



Informatica® SSA-NAME3  
10.5 HotFix 1

# SSA-NAME3 API Reference Guide

Informatica SSA-NAME3 SSA-NAME3 API Reference Guide  
10.5 HotFix 1  
September 2023

© Copyright Informatica LLC 1999, 2023

This software and documentation are provided only under a separate license agreement containing restrictions on use and disclosure. No part of this document may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording or otherwise) without prior consent of Informatica LLC.

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation is subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License.

Informatica and the Informatica logo are trademarks or registered trademarks of Informatica LLC in the United States and many jurisdictions throughout the world. A current list of Informatica trademarks is available on the web at <https://www.informatica.com/trademarks.html>. Other company and product names may be trade names or trademarks of their respective owners.

The information in this documentation is subject to change without notice. If you find any problems in this documentation, report them to us at [infa\\_documentation@informatica.com](mailto:infa_documentation@informatica.com).

Informatica products are warranted according to the terms and conditions of the agreements under which they are provided. INFORMATICA PROVIDES THE INFORMATION IN THIS DOCUMENT "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT.

Publication Date: 2023-11-22

# Table of Contents

<b>Preface .....</b>	<b>15</b>
Informatica Resources. ....	15
Informatica Network. ....	15
Informatica Knowledge Base. ....	15
Informatica Documentation. ....	15
Informatica Product Availability Matrices. ....	16
Informatica Velocity. ....	16
Informatica Marketplace. ....	16
Informatica Global Customer Support. ....	16
<b>Chapter 1: Introduction.....</b>	<b>17</b>
Configure Environment Variables. ....	17
<b>Chapter 2: Program Design.....</b>	<b>19</b>
The Basic API Sequences. ....	19
SSA-NAME3 Sessions. ....	20
Generating a Sample Program. ....	22
<b>Chapter 3: SSA-NAME3 Functions.....</b>	<b>23</b>
Common Parameters. ....	23
Return Value. ....	28
ssan3_connect. ....	29
ssan3_disconnect. ....	29
ssan3_open. ....	29
ssan3_close. ....	30
ssan3_get_keys_encoded. ....	30
ssan3_get_keys. ....	31
ssan3.get_keys_raw(). ....	32
ssan3_convert_keys. ....	33
ssan3_get_ranges_encoded. ....	34
ssan3_get_ranges. ....	35
ssan3.get_ranges_raw(). ....	36
ssan3_convert_ranges. ....	37
ssan3_match_encoded. ....	37
ssan3_match. ....	38
ssan3_get_word_stack. ....	38
ssan3_match_ext. ....	39
ssan3_match_rescore. ....	39
ssan3_info. ....	39
ssan3_info_ext. ....	39

<b>Chapter 4: Language Specific Guidelines.....</b>	<b>40</b>
Data Types. . . . .	40
Language Specific Bindings. . . . .	41
Calling from C. . . . .	41
ssan3_addr_get_cass_field. . . . .	42
ssan3_addr_get_cass_field_cnt. . . . .	42
ssan3_addr_get_cass_field_info. . . . .	43
ssan3_addr_get_del_lines. . . . .	43
ssan3_addr_get_del_lines_ext. . . . .	44
ssan3_addr_get_field. . . . .	45
ssan3_addr_get_field_count. . . . .	46
ssan3_addr_get_field_ext. . . . .	47
ssan3_addr_get_field_idx. . . . .	48
ssan3_addr_get_field_info_ext. . . . .	49
ssan3_addr_get_field_len. . . . .	49
ssan3_addr_get_line_len. . . . .	50
ssan3_addr_get_option. . . . .	51
ssan3_addr_init. . . . .	51
ssan3_addr_parse. . . . .	52
ssan3_addr_preload_country. . . . .	52
ssan3_addr_set_attrib. . . . .	53
ssan3_addr_set_del_lines. . . . .	54
ssan3_addr_set_field_case. . . . .	55
ssan3_addr_set_field_idx. . . . .	55
ssan3_addr_set_field_name. . . . .	56
ssan3_addr_set_lines. . . . .	57
ssan3_addr_set_option. . . . .	58
ssan3_addr_validate. . . . .	58
ssan3_close. . . . .	59
ssan3_connect. . . . .	60
ssan3_convert_keys. . . . .	60
ssan3_convert_ranges. . . . .	61
ssan3_disconnect. . . . .	61
ssan3_errors_get_all. . . . .	61
ssan3_get_keys. . . . .	62
ssan3_get_keys_encoded. . . . .	63
ssan3_get_ranges. . . . .	64
ssan3_get_ranges_encoded. . . . .	64
ssan3_info. . . . .	65
ssan3_info_ext. . . . .	66
ssan3_keys. . . . .	67

ssan3_match. . . . .	68
ssan3_match_encoded. . . . .	69
ssan3_match_ext. . . . .	70
ssan3_match_rescore. . . . .	71
ssan3_open. . . . .	72
ssan3_ranges. . . . .	73
Calling from C#. . . . .	74
Compilation and linking. . . . .	74
addr_get_cass_field. . . . .	75
addr_get_cass_field_cnt. . . . .	75
addr_get_cass_field_info. . . . .	76
addr_get_del_lines. . . . .	76
addr_get_del_lines_ext. . . . .	77
addr_get_field. . . . .	77
addr_get_field_count. . . . .	78
addr_get_field_ext. . . . .	78
addr_get_field_idx. . . . .	79
addr_get_field_info_ext. . . . .	80
addr_get_field_len. . . . .	80
addr_get_line_len. . . . .	81
addr_get_option. . . . .	81
addr_init. . . . .	82
addr_parse. . . . .	82
addr_preload_country. . . . .	83
addr_set_attrib. . . . .	83
addr_set_del_lines. . . . .	84
addr_set_field_case. . . . .	84
addr_set_field_idx. . . . .	85
addr_set_field_name. . . . .	85
addr_set_lines. . . . .	86
addr_set_option. . . . .	87
addr_validate. . . . .	87
close. . . . .	88
convert_keys. . . . .	88
convert_ranges. . . . .	89
disconnect. . . . .	89
errors_get_all. . . . .	89
get_keys. . . . .	90
get_keys_encoded. . . . .	90
get_ranges. . . . .	91
get_ranges_encoded. . . . .	92
info. . . . .	93

keys. . . . .	93
match. . . . .	94
match_encoded. . . . .	95
open. . . . .	95
ranges. . . . .	96
Calling from C++. . . . .	97
Use of Destructors. . . . .	97
ssan3_addr_get_cass_field. . . . .	97
ssan3_addr_get_cass_field_cnt. . . . .	98
ssan3_addr_get_cass_field_info. . . . .	98
ssan3_addr_get_del_lines. . . . .	99
ssan3_addr_get_del_lines_ext. . . . .	100
ssan3_addr_get_field. . . . .	101
ssan3_addr_get_field_count. . . . .	102
ssan3_addr_get_field_ext. . . . .	102
ssan3_addr_get_field_idx. . . . .	103
ssan3_addr_get_field_info_ext. . . . .	104
ssan3_addr_get_field_len. . . . .	105
ssan3_addr_get_line_len. . . . .	105
ssan3_addr_get_option. . . . .	106
ssan3_addr_init. . . . .	106
ssan3_addr_parse. . . . .	107
ssan3_addr_preload_country. . . . .	107
ssan3_addr_set_attrib. . . . .	108
ssan3_addr_set_del_lines. . . . .	108
ssan3_addr_set_field_case. . . . .	109
ssan3_addr_set_field_idx. . . . .	110
ssan3_addr_set_field_name. . . . .	110
ssan3_addr_set_lines. . . . .	111
ssan3_addr_set_option. . . . .	112
ssan3_addr_validate. . . . .	113
ssan3_close. . . . .	113
ssan3_connect. . . . .	114
ssan3_convert_keys. . . . .	114
ssan3_convert_ranges. . . . .	115
ssan3_disconnect. . . . .	115
ssan3_errors_get_all. . . . .	116
ssan3_get_keys. . . . .	116
ssan3_get_keys_encoded. . . . .	117
ssan3_get_ranges. . . . .	118
ssan3_get_ranges_encoded. . . . .	119
ssan3_info. . . . .	120

ssan3_keys. . . . .	120
ssan3_match. . . . .	121
ssan3_match_encoded. . . . .	122
ssan3_open. . . . .	123
ssan3_ranges. . . . .	124
Calling from COBOL. . . . .	125
SSAN3-ADDR-GET-CASS-FIELD. . . . .	125
SSAN3-ADDR-GET-CASS-FIELD-CNT. . . . .	125
SSAN3-ADDR-GET-CASS-FIELD-INFO. . . . .	126
SSAN3-ADDR-GET-DEL-LINES. . . . .	126
SSAN3-ADDR-GET-DEL-LINES-EXT. . . . .	127
SSAN3-ADDR-GET-FIELD. . . . .	128
SSAN3-ADDR-GET-FIELD-COUNT. . . . .	129
SSAN3-ADDR-GET-FIELD-EXT. . . . .	129
SSAN3-ADDR-GET-FIELD-IDX. . . . .	130
SSAN3-ADDR-GET-FIELD-INFO-EXT. . . . .	131
SSAN3-ADDR-GET-FIELD-LEN. . . . .	131
SSAN3-ADDR-GET-LINE-LEN. . . . .	132
SSAN3-ADDR-GET-OPTION. . . . .	132
SSAN3-ADDR-INIT. . . . .	133
SSAN3-ADDR-PARSE. . . . .	133
SSAN3-ADDR-PRELOAD-COUNTRY. . . . .	134
SSAN3-ADDR-SET-ATTRIB. . . . .	135
SSAN3-ADDR-SET-DEL-LINES. . . . .	135
SSAN3-ADDR-SET-FIELD-CASE. . . . .	136
SSAN3-ADDR-SET-FIELD-IDX. . . . .	136
SSAN3-ADDR-SET-FIELD-NAME. . . . .	137
SSAN3-ADDR-SET-LINES. . . . .	137
SSAN3-ADDR-SET-OPTION. . . . .	138
SSAN3-ADDR-VALIDATE. . . . .	139
SSAN3-CLOSE. . . . .	139
SSAN3-CONNECT. . . . .	140
SSAN3-CONVERT-KEYS. . . . .	140
SSAN3-CONVERT-RANGES. . . . .	141
SSAN3-DISCONNECT. . . . .	141
SSAN3-ERRORS-GET-ALL. . . . .	142
SSAN3-GET-KEYS. . . . .	142
SSAN3-GET-RANGES. . . . .	143
SSAN3-INFO. . . . .	144
SSAN3-KEYS. . . . .	145
SSAN3-MATCH. . . . .	145
SSAN3-OPEN. . . . .	146

SSAN3-RANGES. . . . .	147
Calling from Java. . . . .	148
Coding the ssan3_close Call outside of the Finalize() Method. . . . .	148
ssan3_addr_get_cass_field. . . . .	148
ssan3_addr_get_cass_field_cnt. . . . .	149
ssan3_addr_get_cass_field_info. . . . .	150
ssan3_addr_get_del_lines. . . . .	150
ssan3_addr_get_del_lines_ext. . . . .	151
ssan3_addr_get_field. . . . .	152
ssan3_addr_get_field_count. . . . .	153
ssan3_addr_get_field_ext. . . . .	153
ssan3_addr_get_field_idx. . . . .	154
ssan3_addr_get_field_info_ext. . . . .	155
ssan3_addr_get_field_len. . . . .	155
ssan3_addr_get_line_len. . . . .	156
ssan3_addr_get_option. . . . .	156
ssan3_addr_init. . . . .	157
ssan3_addr_parse. . . . .	158
ssan3_addr_preload_country. . . . .	158
ssan3_addr_set_attrib. . . . .	159
ssan3_addr_set_del_lines. . . . .	159
ssan3_addr_set_field_case. . . . .	160
ssan3_addr_set_field_idx. . . . .	161
ssan3_addr_set_field_name. . . . .	161
ssan3_addr_set_lines. . . . .	162
ssan3_addr_set_option. . . . .	163
ssan3_addr_validate. . . . .	163
ssan3_close. . . . .	164
ssan3_convert_keys. . . . .	164
ssan3_convert_ranges. . . . .	165
ssan3_disconnect. . . . .	165
ssan3_errors_get_all. . . . .	166
ssan3_get_keys. . . . .	166
ssan3_get_keys_encoded. . . . .	167
ssan3_get_ranges. . . . .	168
ssan3_get_ranges_encoded. . . . .	168
ssan3_info. . . . .	169
ssan3_info_ext. . . . .	170
ssan3_keys. . . . .	171
ssan3_match. . . . .	172
ssan3_match_encoded. . . . .	173
ssan3_match_ext. . . . .	174



ssan3_match_rescore. . . . .	175
ssan3_open. . . . .	176
ssan3_ranges. . . . .	176
Calling from Microsoft SQL Server. . . . .	177
Prerequisites. . . . .	177
ssan3xp_addr_get_cass_field. . . . .	178
ssan3xp_addr_get_cass_field_cnt. . . . .	178
ssan3xp_addr_get_cass_field_info. . . . .	179
ssan3xp_addr_get_del_lines. . . . .	180
ssan3xp_addr_get_del_lines_ext. . . . .	181
ssan3xp_addr_get_field. . . . .	182
ssan3xp_addr_get_field_count. . . . .	182
ssan3xp_addr_get_field_ext. . . . .	183
ssan3xp_addr_get_field_idx. . . . .	184
ssan3xp_addr_get_field_info_ext. . . . .	184
ssan3xp_addr_get_field_len. . . . .	185
ssan3xp_addr_get_line_len. . . . .	186
ssan3xp_addr_get_option. . . . .	186
ssan3xp_addr_init. . . . .	187
ssan3xp_addr_parse. . . . .	187
ssan3xp_addr_preload_country. . . . .	188
ssan3xp_addr_set_attrib. . . . .	189
ssan3xp_addr_set_del_lines. . . . .	189
ssan3xp_addr_set_field_case. . . . .	190
ssan3xp_addr_set_field_idx. . . . .	191
ssan3xp_addr_set_field_name. . . . .	191
ssan3xp_addr_set_lines. . . . .	192
ssan3xp_addr_set_option. . . . .	193
ssan3xp_addr_validate. . . . .	193
ssan3xp_close. . . . .	194
ssan3xp_connect. . . . .	194
ssan3xp_convert_keys. . . . .	195
ssan3xp_convert_ranges. . . . .	196
ssan3xp_disconnect. . . . .	196
ssan3xp_errors_get_all. . . . .	197
get_keys. . . . .	197
ssan3xp_get_keys_encoded_text. . . . .	198
ssan3xp_get_keys_encoded_unicode. . . . .	199
ssan3xp_get_ranges. . . . .	200
get_ranges_encoded. . . . .	201
ssan3xp_get_ranges_encoded_unicode. . . . .	202
ssan3xp_info. . . . .	203

ssan3xp_keys. . . . .	203
ssan3xp_match. . . . .	205
ssan3xp_match_encoded_text. . . . .	205
ssan3xp_match_encoded_unicode. . . . .	206
ssan3xp_open. . . . .	207
ssan3xp_ranges. . . . .	208
Calling from Perl. . . . .	209
addr_get_cass_field. . . . .	209
addr_get_cass_field_cnt. . . . .	209
addr_get_cass_field_info. . . . .	210
addr_get_del_lines. . . . .	210
addr_get_del_lines_ext. . . . .	211
addr_get_field. . . . .	212
addr_get_field_count. . . . .	212
addr_get_field_ext. . . . .	213
addr_get_field_idx. . . . .	214
addr_get_field_info_ext. . . . .	214
addr_get_field_len. . . . .	215
addr_get_line_len. . . . .	215
addr_get_option. . . . .	216
addr_init. . . . .	216
addr_parse. . . . .	217
addr_preload_country. . . . .	217
addr_set_attrib. . . . .	218
addr_set_del_lines. . . . .	218
addr_set_field_case. . . . .	219
addr_set_field_idx. . . . .	220
addr_set_field_name. . . . .	220
addr_set_lines. . . . .	221
addr_set_option. . . . .	222
addr_validate. . . . .	222
close. . . . .	222
convert_keys. . . . .	223
convert_ranges. . . . .	223
disconnect. . . . .	224
errors_get_all. . . . .	224
get_keys_encoded. . . . .	225
get_ranges. . . . .	225
info. . . . .	226
keys. . . . .	227
match. . . . .	227
match_encoded. . . . .	228

ranges. . . . .	229
Calling from Visual Basic 6. . . . .	230
addr_get_cass_field. . . . .	230
ssan3_addr_get_cass_field_cnt. . . . .	230
ssan3_addr_get_cass_field_info. . . . .	231
ssan3_addr_get_del_lines. . . . .	232
ssan3_addr_get_del_lines_ext. . . . .	232
ssan3_addr_get_field. . . . .	233
ssan3_addr_get_field_count. . . . .	234
ssan3_addr_get_field_ext. . . . .	235
ssan3_addr_get_field_idx. . . . .	235
ssan3_addr_get_field_info_ext. . . . .	236
ssan3_addr_get_field_len. . . . .	237
ssan3_addr_get_line_len. . . . .	237
ssan3_addr_get_option. . . . .	238
ssan3_addr_init. . . . .	238
ssan3_addr_parse. . . . .	239
ssan3_addr_preload_country. . . . .	239
ssan3_addr_set_attrib. . . . .	240
ssan3_addr_set_del_lines. . . . .	240
ssan3_addr_set_field_case. . . . .	241
ssan3_addr_set_field_idx. . . . .	242
ssan3_addr_set_field_name. . . . .	242
ssan3_addr_set_lines. . . . .	243
ssan3_addr_set_option. . . . .	244
ssan3_addr_validate. . . . .	244
ssan3_close. . . . .	245
ssan3_connect. . . . .	246
ssan3_convert_keys. . . . .	246
ssan3_convert_ranges. . . . .	247
ssan3_disconnect. . . . .	247
ssan3_errors_get_all. . . . .	247
ssan3_get_keys. . . . .	248
ssan3_get_keys_encoded. . . . .	249
ssan3_get_ranges. . . . .	249
ssan3_get_ranges_encoded. . . . .	250
ssan3_info. . . . .	251
ssan3_keys. . . . .	252
ssan3_match. . . . .	253
ssan3_match_encoded. . . . .	253
ssan3_open. . . . .	254
ssan3_ranges. . . . .	255

Calling from Visual Basic .NET. . . . .	256
addr_get_cass_field. . . . .	257
addr_get_field_count. . . . .	257
addr_get_cass_field_info. . . . .	257
addr_get_del_lines. . . . .	258
addr_get_del_lines_ext. . . . .	259
addr_get_field. . . . .	259
addr_get_field_count. . . . .	260
addr_get_field_ext. . . . .	260
addr_get_field_idx. . . . .	261
addr_get_field_info_ext. . . . .	262
addr_get_field_len. . . . .	262
addr_get_line_len. . . . .	263
addr_get_option. . . . .	263
addr_init. . . . .	264
addr_parse. . . . .	264
addr_preload_country. . . . .	265
addr_set_attrib. . . . .	265
addr_set_del_lines. . . . .	266
addr_set_field_case. . . . .	266
addr_set_field_idx. . . . .	267
addr_set_field_name. . . . .	267
addr_set_lines. . . . .	268
addr_set_option. . . . .	269
addr_validate. . . . .	269
close. . . . .	270
ssan3_convert_keys. . . . .	270
ssan3_convert_ranges. . . . .	271
disconnect. . . . .	271
errors_get_all. . . . .	272
get_keys. . . . .	272
ssan3_get_keys_encoded. . . . .	273
get_ranges. . . . .	273
ssan3_get_ranges_encoded. . . . .	274
ssan3_info. . . . .	275
keys. . . . .	275
match. . . . .	276
ssan3_match_encoded. . . . .	277
open. . . . .	278
ranges. . . . .	278
<b>Chapter 5: Controls. . . . .</b>	<b>280</b>
ssan3_connect Controls. . . . .	280

ssan3_disconnect Controls. . . . .	280
ssan3_open Controls. . . . .	280
ssan3_close Controls. . . . .	281
ssan3_get_keys_encoded Controls. . . . .	282
ssan3_convert_keys Controls. . . . .	285
ssan3_get_ranges_encoded Controls. . . . .	285
ssan3_convert_ranges Controls. . . . .	290
ssan3_match_encoded Controls. . . . .	290
ssan3_get_word_stack. . . . .	309
ssan3_info Controls. . . . .	310

## **Chapter 6: Advanced Controls. . . . . 313**

UNICODE_ENCODING. . . . .	313
PURPOSE. . . . .	315
SSA-NAME3 Error Messages. . . . .	320

## **Chapter 7: Composite Keys. . . . . 322**

Combining multiple fields into one key. . . . .	322
Key Sizes. . . . .	323
Limitations. . . . .	323
Performance. . . . .	323
Extended Functionality for Composite Key. . . . .	324
Keywords. . . . .	324
Updated Controls. . . . .	324
Useful Key and Range Keywords for Composite Keys. . . . .	325
Example. . . . .	326
SSA-NAME3 Error Messages. . . . .	327

## **Chapter 8: Address Standardization. . . . . 330**

Initialization. . . . .	330
Character Sets and Countries. . . . .	330
Providing an Input Address. . . . .	341
Parsing an Address. . . . .	341
Validating an Address. . . . .	343
Retrieving Address Fields. . . . .	345
Setting Options. . . . .	347
Getting Options. . . . .	348
Sample Code. . . . .	348
Validation Database Files. . . . .	349

## **Chapter 9: ASM Workbench. . . . . 353**

Introduction. . . . .	353
Launching the ASM Workbench. . . . .	353

ASM Workbench Input Options. . . . .	354
Country Specific Input. . . . .	354
Character Set. . . . .	355
Country Preload Option. . . . .	355
Address Input Type. . . . .	355
Options. . . . .	356
Parsing and Validation Frame. . . . .	356
Attributes. . . . .	356
Suggested Address Label Display. . . . .	357
Address Result Panel Display. . . . .	358
Validation Status and Database Version Display. . . . .	358
Output Result Frame Column Selection Menu. . . . .	360
Field Status Display. . . . .	361
CASS Field Status Display. . . . .	363
CASS Summary Report Display. . . . .	364
Statistics Reports - CASS Certification. . . . .	365
File Menu Options. . . . .	365
ASM Workbench and Batch Test utility. . . . .	366
<b>Index. . . . .</b>	<b>368</b>

# Preface

Refer to the Informatica SSA-NAME3 API Reference Guide to learn about the typical program process flow for building an identity search application. This reference guide also describes the API functions, the parameters required by these functions, and the information returned.

## Informatica Resources

Informatica provides you with a range of product resources through the Informatica Network and other online portals. Use the resources to get the most from your Informatica products and solutions and to learn from other Informatica users and subject matter experts.

### Informatica Network

The Informatica Network is the gateway to many resources, including the Informatica Knowledge Base and Informatica Global Customer Support. To enter the Informatica Network, visit <https://network.informatica.com>.

As an Informatica Network member, you have the following options:

- Search the Knowledge Base for product resources.
- View product availability information.
- Create and review your support cases.
- Find your local Informatica User Group Network and collaborate with your peers.

### Informatica Knowledge Base

Use the Informatica Knowledge Base to find product resources such as how-to articles, best practices, video tutorials, and answers to frequently asked questions.

To search the Knowledge Base, visit <https://search.informatica.com>. If you have questions, comments, or ideas about the Knowledge Base, contact the Informatica Knowledge Base team at [KB\\_Feedback@informatica.com](mailto:KB_Feedback@informatica.com).

### Informatica Documentation

Use the Informatica Documentation Portal to explore an extensive library of documentation for current and recent product releases. To explore the Documentation Portal, visit <https://docs.informatica.com>.

If you have questions, comments, or ideas about the product documentation, contact the Informatica Documentation team at [infa\\_documentation@informatica.com](mailto:infa_documentation@informatica.com).

## Informatica Product Availability Matrices

Product Availability Matrices (PAMs) indicate the versions of the operating systems, databases, and types of data sources and targets that a product release supports. You can browse the Informatica PAMs at <https://network.informatica.com/community/informatica-network/product-availability-matrices>.

## Informatica Velocity

Informatica Velocity is a collection of tips and best practices developed by Informatica Professional Services and based on real-world experiences from hundreds of data management projects. Informatica Velocity represents the collective knowledge of Informatica consultants who work with organizations around the world to plan, develop, deploy, and maintain successful data management solutions.

You can find Informatica Velocity resources at <http://velocity.informatica.com>. If you have questions, comments, or ideas about Informatica Velocity, contact Informatica Professional Services at [ips@informatica.com](mailto:ips@informatica.com).

## Informatica Marketplace

The Informatica Marketplace is a forum where you can find solutions that extend and enhance your Informatica implementations. Leverage any of the hundreds of solutions from Informatica developers and partners on the Marketplace to improve your productivity and speed up time to implementation on your projects. You can find the Informatica Marketplace at <https://marketplace.informatica.com>.

## Informatica Global Customer Support

You can contact a Global Support Center through the Informatica Network or by telephone.

To find online support resources on the Informatica Network, click **Contact Support** in the Informatica Intelligent Cloud Services Help menu to go to the **Cloud Support** page. The **Cloud Support** page includes system status information and community discussions. Log in to Informatica Network and click **Need Help** to find additional resources and to contact Informatica Global Customer Support through email.

The telephone numbers for Informatica Global Customer Support are available from the Informatica web site at <https://www.informatica.com/services-and-training/support-services/contact-us.html>.



# CHAPTER 1

## Introduction

This document is intended to be the Application Developer's resource for assistance with using the SSA-NAME3 API Functions.

The API gives user developed application programs the ability to use the functionality within SSANAME3. The API is available with wrappers for Active X as well as the standard interfaces for C/C++, COBOL and Java.

The API supports both local calls to the SSA-NAME3 Callable Routine (DLL, Shared Library, Load Module etc), as well as socket calls (through TCP/IP) to a server implementation of the Callable Routine.

The ability to call the SSA-NAME3 Server from a remote client has been provided so that a program may be developed and tested on one platform (example, a PC) while having its SSA-NAME3 calls executed on another platform (example, a server). This may increase the flexibility of the development and testing phases.

**Note:** In production, the calling application and the SSA-NAME3 Callable Routine should reside on the same computer, or severe performance degradation may be expected.

## Configure Environment Variables

When you implement SSA-NAME3 by using a DLL, shared library, or load module, configure the following environment variables:

### SSAPR

Specifies the absolute path for the standard populations. You can find the standard populations in the following directory:

- On Windows: <Informatica Installation Directory>\pr
- On UNIX: <Informatica Installation Directory>/pr

You can override the SSAPR environment variable at run time by specifying the SSAPR control in a `ssan3_open` call.

### PATH

Specifies the search path for executable programs and libraries. Include the absolute path for the SSA-NAME3 binary directory in the PATH environment variable.

For example, `C:\InformaticaIR\bin`.

### SSA\_LIHOST

Specifies the host name of the License Server and the port number on which the License Server listens.

For example, `localhost:1660`.

**Note:** Some populations might create temporary files for sorting while loading populations into the memory. If the `SSAWORKDIR` environment variable is set, a directory named `Process ID_temp` is created. `SSA-NAME3` does not clean up the temporary files automatically. You need to manually remove the directory after the process exits.

## CHAPTER 2

# Program Design

For information about the general design of the application programs that will need to call SSANAME3, refer to the *APPLICATION and DATABASE DESIGN GUIDE* sections:

- The Basic Process Flow
- The Basic Function Flow

## The Basic API Sequences

An application using SSA-NAME3 Version 2 will normally make 5 API calls:

1. **ssan3\_open()** to open a session to the SSA-NAME3 services
2. **ssan3\_close()** to terminate an open session to SSA-NAME3
3. **ssan3\_get\_keys\_encoded()** to build keys on names or addresses
4. **ssan3\_get\_ranges\_encoded()** to get search ranges for names or addresses
5. **ssan3\_match\_encoded()** to compare and match two records

In addition, if using the SSA-NAME3 Server, it is necessary to make the following two calls to open and close the socket connections:

- **ssan3\_connect()**
- **ssan3\_disconnect()**

If used, the **ssan3\_connect** call should be done prior to the **ssan3\_open** call, and the **ssan3\_disconnect** call done after the **ssan3\_close** call.

Only one connect, open, close and disconnect call should be made during the active life of the application or thread, that is from when the time the application or thread is started to the time it is terminated. These calls should frame the rest of the work done by the application.

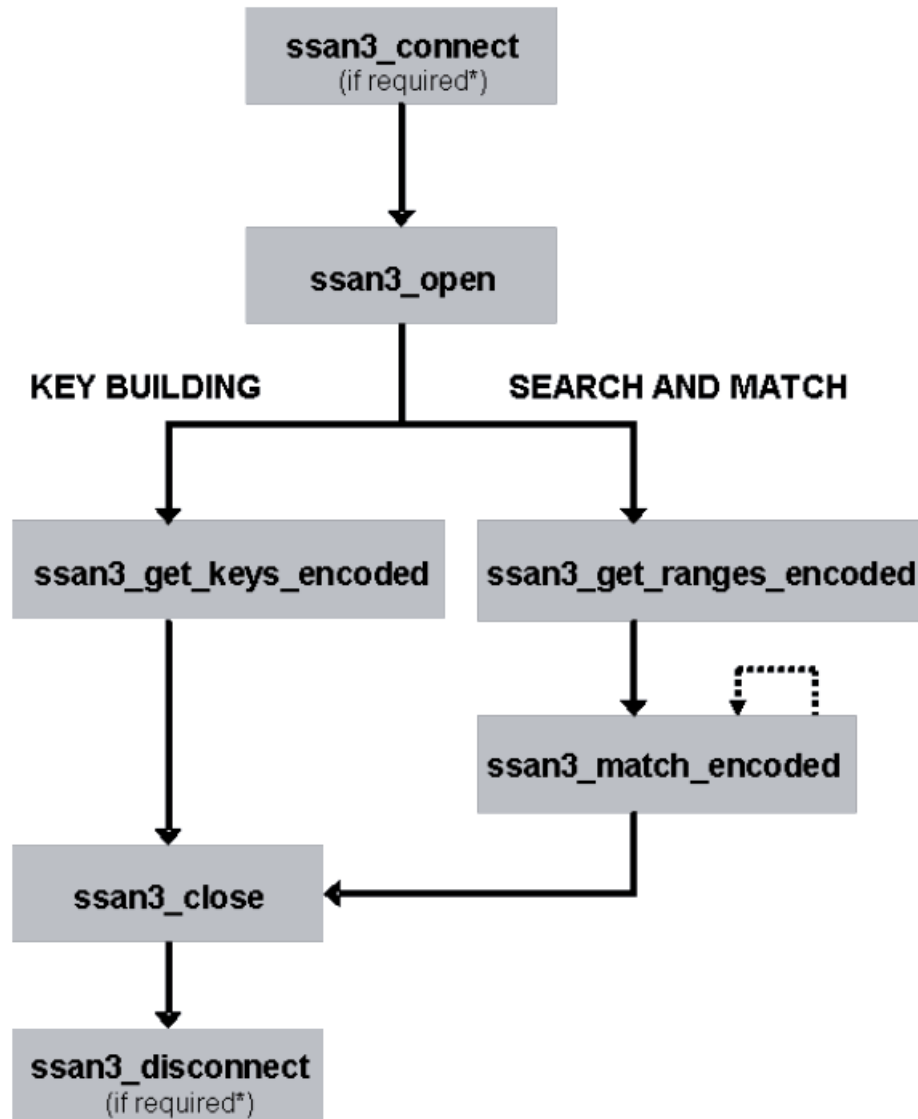
It is imperative that socket-handles and Session-IDs are not shared across concurrent transactions or threads.

In addition, if using the SSA-NAME3 Server and you wish to use 5-byte binary keys in order to save on disk space, it is necessary to make the following two calls to convert the 8-byte keys passed from the SSA-NAME3 Server:

- **ssan3\_convert\_keys()**
- **ssan3\_convert\_ranges()** If used, the **ssan3\_convert\_keys** call should be done after the **ssan3\_get\_keys\_encoded** call, and the **ssan3\_convert\_ranges** call done after the **ssan3\_get\_ranges\_encoded** call.

Note that some databases, programming languages and transport mechanisms will not support binary keys. If you are unsure, you need to contact Informatica Corporation.

At a very high level, the sequence of API's in an application using SSA-NAME3 is depicted as shown below:



## SSA-NAME3 Sessions

SSA-NAME3 manages the resources it needs to satisfy API call requests in memory areas called "sessions". SSA-NAME3 acquires, manages and releases these memory areas.

A session is established explicitly, via an **ssan3\_open** call, or implicitly, via any other call if SSANAME3 does not recognize the caller.

The establishment of a session goes through an initialization process. During this process, the specified Population Rules are loaded, and a work-area for SSA-NAME3's use is allocated. This is work that should happen as infrequently as possible, and this is to some extent under the control of the application designer.

It is important for performance that all of the function calls from the same transaction or process use the same session. In other words, calls to SSA-NAME3 from the same transaction or process for **ssan3\_get\_keys\_encoded**, **ssan3\_get\_ranges\_encoded**, and **ssan3\_match\_encoded**, will perform faster and use fewer resources if the same session is used during these calls.

One copy of SSA-NAME3 can handle 1024 sessions. In some cases, a large number of concurrent users, a large number of inactive sessions, or a large number of calls that do not properly re-use sessions, could lead to SSA-NAME3 running out of sessions.

If it is truly a large number of real concurrent users, then another copy of SSA-NAME3 may need to be started/loaded.

Otherwise, SSA-NAME3 will do its best to reuse sessions intelligently. However, if all else fails the SSA-NAME3 instance may need to be unloaded/reloaded or the SSA-NAME3 server re-started.

If a large number of users are expected to be using SSA-NAME3 services, it is recommended that the application be designed as a server process that manages a pool of available sessions for calling client applications.

### Explicit Sessions

As said previously, it is recommended that an application open explicit sessions, as this provides more control over the system's resources and performance.

To request an explicit session, an **ssan3\_open** call should be done passing a Session-ID of -1. This will cause a new valid Session-ID to be allocated and returned to the calling program (example, Session-ID 1048576), providing SSA-NAME3 with a handle to the initialized resources. It is this Session-ID (example, 1048576) that should be used for all subsequent SSA-NAME3 calls in the current transaction/ process in order for it to use the same session.

Do not pass a value of "-1" in the Session-ID field of a function unless you wish to force SSA-NAME3 to do an implicit **ssan3\_open** again.

A session allocated through an **ssan3\_open** call is locked and cannot be re-used by another session until released through the **ssan3\_close** call, or by exceeding an inactivity time-out value as discussed below.

Once a session is finished with, it should be released through the **ssan3\_close** call using the allocated Session-ID (example, 1048576). The Session-ID is then set to -1 by the **ssan3\_close** call.

If it is likely to be common that sessions will be left open and become inactive, without being explicitly closed, then it is possible to control the length of time a session is held by SSA-NAME3. An inactivity time-out value (in seconds) may be passed in the **ssan3\_open** call, or if using the SSA-NAME3 server, may be specified as a server start-up parameter. Once a period of inactivity has reached the time-out value, the session may be closed and reallocated to another caller. If the original caller tries to later reuse that session, it will be allocated a new Session-ID and cause the initialization process to be repeated.

### Ephemeral Sessions

If an explicit **ssan3\_open** call is not made, an implicit or "ephemeral" session is initiated for the calling application. An ephemeral session will be created if:

- The session id passed in any call is not known to SSA-NAME3
- The **ssan3\_open** call does not pass a Session-ID value of -1

While ephemeral opens save the application from issuing the **ssan3\_open** call, if SSA-NAME3 at some stage runs out of session handles, it may allocate a new session to an existing ephemeral session slot.

This means that the next call from the application who's ephemeral session was re-used will cause the initialization process to be repeated, and may lead to some performance degradation.

SSA-NAME3 will overwrite ephemeral sessions before checking for timed-out sessions.

Once a session is finished with, it should be released through the **ssan3\_close** call.

# Generating a Sample Program

It is recommended that the Sample Program feature of the Developer's Workbench be used to view language specific code examples. This is because the parameters required for each function, and the way they are specified, may vary depending on the programming language used.

## CHAPTER 3

# SSA-NAME3 Functions

This section describes the available SSA-NAME3 V2 API functions.

The calling conventions and parameters described in this section are intended to be a generic guide to programming. For programming language specific guidelines, refer to the Language Specific Guidelines chapter in conjunction after first familiarizing yourself with the ideas in this chapter.

## Common Parameters

The following parameters are common to many of the API Functions.

### Socket Handle

Only used if your application is calling the SSA-NAME3 Server, otherwise it should be set to `-1`. If using the SSA-NAME3 server, the `ssan3_connect` call will allocate a valid socket handle which should then be used on all subsequent calls.

### Session ID

Should be either `-1` on the `ssan3_open` call, if opening a new session, or a valid Session ID (as returned from `ssan3_open`) on any other call. For more information on Sessions, refer to SSA-NAME3 Sessions.

### System

Qualifies and defines the location of the Population Rules to be used in the call. It corresponds to the name of the sub-directory (within the `pr` directory), or low-level extension of the PR dataset in z/OS environments, in which the Population Files (YSPs, YCPs or YLPs) are stored.

### Population

The name of the Population rule set to be used. This generally corresponds to a Country/Language rule set (example, Australia, Brazil, UK, and USA); however, it could be any set of rules, especially if using a Custom Population (example, English Song Titles, or USA OFAC). Use the Population that corresponds to the data you are using. SSA-NAME3 will search for the Population in the location defined by the `SSAPR` environment variable and System parameter. It will search for Populations in the following order:

- `Population.ylp` (Local Population)
- `Population.ycp` (Custom Population)

- `Population.yzp` (Standard Population)

## Controls

When calling many of the API functions, it is necessary to specify "Controls" to be used. These Controls define and refine the behaviour of the function. The format and content of the Controls will be dependent on the function being called and are described in the Controls section.

## Response Code

Indicates the success or failure of a call to SSA-NAME3. A Response Code value of zero indicates a successful call. If the Response Code is not zero, then a description of the problem will be reported in the Error Message parameter.

## Error Message

Refer to the section SSA-NAME3 Error Messages to see a list of possible messages.

## Data

The SSA-NAME3 calls **ssan3\_get\_keys\_encoded**, **ssan3\_get\_ranges\_encoded** and **ssan3\_match\_encoded** all require data to work on. This is the user's identity data and will take the form of names, addresses, ID numbers, dates and other attributes. The name of the parameter for **ssan3\_get\_keys\_encoded** and **ssan3\_get\_ranges\_encoded** is Key Field Data, and for the **ssan3\_match\_encoded** call is Search Data and File Data.

Key Field Data contains the field value(s) to be used for the building of the SSA-NAME3 Keys when using the **ssan3\_get\_keys\_encoded** function and for building search ranges when using the **ssan3\_get\_ranges\_encoded** function.

Search Data contains the field value(s) from the search or transaction record (example, taken from a screen or input file) to be used to compare against the File data in an **ssan3\_match\_encoded** call.

File Data contains the field value(s) from the file record (example, the candidate record) to be used to compare against the Search data in an **ssan3\_match\_encoded** call.

The data fields required for each Key-Field or Match Purpose are shown in the Developer's Workbench for your Population.

Any field can be repeated. An example of a repeating field might be passing a person's current name and their former name in the one call for key-building.

The data can be provided by the calling application in either of two formats:

### Tagged Data Format (the default)

The Tagged Data Format is the default method of supplying data in the **ssan3\_get\_keys\_encoded**, **ssan3\_get\_ranges\_encoded**, **ssan3\_get\_word\_stack**, and **ssan3\_match\_encoded** function calls.

By using the Tagged Data Format, the offsets and lengths of the data fields being passed do not need to be specified. Instead, a notation of field-names and delimiters is used to break up the fields. The field names are reserved words. The field names available for your Population rule-set can be found in the Developer's Workbench.

Note the use of the 3 delimiters to terminate the data. For example, if building keys for a Person's Name, the format would be similar to:

```
*Person_Name*Jane Smith***
```

Where `Person_Name` is the field name, the "\*" is the delimiter, and "Jane Smith" is user-supplied data.

An example of building keys on a repeating field (in this case, current name and former name):

```
*Person_Name*Jane Smith*Person_Name*Jane Brown***
```



The following example shows the data supplied to an **ssan3\_match** call for a Match Purpose that is using a person's name and address. The tagged data format for the Search and File Data will look like:

```
*Person_Name*Andrew Barker*Address_Part1*23 Acacia Drive
Sheffield*Address_Part2*Yorkshire
```

Where `Person_Name` and `Address_Part1` are the field names, the "\*" is the delimiter, and the rest is user-supplied data.

A special format exists for Filters used in Multi-Purpose matching. This format is as follows:

```
*Filtern*<single value>
```

or

```
*Filtern(List(value1,value2,... valuen))
```

For more information on Filters, refer to the Controls section for the **ssan3\_match** call.

By default the delimiter is an asterisk, however, it can be overridden by using the `DELIMITER=` Control in the appropriate function call.

If your data can contain asterisks, make sure that these are either cleaned out prior to calling the `SSA-NAME3` functions, or use a different `DELIMITER=` setting.

**Note:** It is possible to pass redundant data as part of the Tagged Data Format string, meaning that the application program can make use of the same structure when calling **ssan3\_get\_keys**, **ssan3\_get\_ranges** or **ssan3\_match**.

#### Scatter/Gather Format

A Scatter/Gather data format can also be used to supply data with the **ssan3\_get\_keys\_encoded**, **ssan3\_get\_ranges\_encoded**, **ssan3\_get\_word\_stack**, and **ssan3\_match\_encoded** function calls.

There are three formats:

```
Offset1,Length1,...,Offsetn,Lengthn
```

and

```
Field1,Offset1,Length1,...,Fieldn,Offsetn,Lengthn
Field1,Offset1,Length1,Encoding1... ,Fieldn,Offsetn,Lengthn,Encodingn
```

The first format may be used when passing Key Field Data to the **ssan3\_get\_keys\_encoded** and **ssan3\_get\_ranges\_encoded** calls. The second format must be used when passing Search Data and File Data to the **ssan3\_match\_encoded** call, and may also be used to pass Key Field Data. The third format can be used when passing Search Data and File Data to the **ssan3\_match\_encoded** call, and may also be used to pass Key Field Data.

**Note:** Lengths should be specified as number of bytes, not number of characters. This is important when passing data in multi-byte character sets such as Japanese or Unicode.

Following are all valid scatter / gather examples for passing Key Field Data:

```
FIELD=Person_Name Layout=0,50
```

Passes a single name, to be found at offset 0 in the Key Field Data parameter, for a length of 50.

```
FIELD=Person_Name Layout=0,50,50,50
```

Passes two names (example, a current name and a former name). The first name is found at offset 0 in the Key Field Data parameter, for a length of 50. The second name is found at offset 50 in the Key Field Data, also for a length 50.

```
FIELD=Person_Name Layout=Person_Name,0,50
```

Passes a single name, to be found at offset 0 in the Key Field Data parameter, for a length of 50.

```
FIELD=Person_Name
Layout=Person_Name,0,50,Person_Name,50,50
```

Passes two names (example, a current name and a former name). The first name is found at offset 0 in the Key Field Data parameter, for a length of 50. The second name is found at offset 50 in the Key Field Data, also for a length 50.

```
FIELD=Person_Name Layout=Person_Name,0,50,
Address_Part1,50,50
```

Passes a single name, to be found at offset 0 in the Key Field Data parameter, for a length of 50. Address\_Part1 is ignored.

The following example shows the valid scatter / gather formats for passing Search and File Data to the **ssan3\_match\_encoded** call:

```
PURPOSE=Resident Search=Person_Name,0,50,
Address_Part1,50,100,Address_Part2,150,50
File=Person_Name,0,50,Address_Part1,50,100,
Address_Part2,150,50
```

Passes a single name, to be found at offset 0 in the Key Field Data parameter, for a length of 50, a single street address, to be found at offset 50, for a length of 100, and a single locality line, to be found at offset 150, for a length of 50.

```
PURPOSE=Resident Search=Person_Name,0,50,
Person_Name,200,50,Address_Part1,50,100,
Address_Part2,150,50 File=Person_Name,0,50,
Person_Name,200,50,Address_Part1,50,100,
Address_Part2,150,50
```

Passes a single name (example, current name), to be found at offset 0 in the Key Field Data parameter, for a length of 50, a single street address, to be found at offset 50, for a length of 100, a single locality line, to be found at offset 150, for a length of 50, and another name (e.g. a former name), to be found at offset 200, for a length of 50.

```
PURPOSE=Resident Search=Organization_Name,0,50,
Person_Name,50,50,Address_Part1,100,100,
Address_Part2,200,50 File=Organization_Name,0,50,
Person_Name,50,50,Address_Part1,100,100,
Address_Part2,200,50
```

Passes a single name, to be found at offset 50 in the Key Field Data parameter, for a length of 50, a single street address, to be found at offset 100, for a length of 100, and a single locality line, to be found at offset 200, for a length of 50.

**Note:** Organization\_Name is ignored as it is not used in the Resident Purpose.

```
PURPOSE=Resident Search=Organization_Name,0,50,8,
Person_Name,50,50,8,Address_Part1,100,100,8,
Address_Part2,200,50,8 File=Organization_Name,0,50,8,
Person_Name,50,50,8,Address_Part1,100,100,8,
Address_Part2,200,50,8
```

This is the same as the previous example except it also specifies the encoding for each field. In this example the encoding is '8' for UTF-8. The encoding can be different for each field. The values are as described for the UNICODE\_ENCODING Control.

By using the syntax that includes the field name, it is possible to pass redundant fields which will be ignored by the function. The function will only access the fields needed by the Key Field Data or Match Purpose. In this way, the calling application can use a common data structure for **ssan3\_get\_keys\_encoded**, **ssan3\_get\_ranges\_encoded** and **ssan3\_match\_encoded**.

It is highly recommend that the scatter/gather format be used in all cases where the data is not in the same character set.

#### Field Limit

Whichever format is used to pass the data, either Tagged or Scatter/Gather, the maximum number of fields which may be passed in a single call is 256.

#### Data Limitations

The length for each field in the scatter/gather or tagged format is 2048. There is no limit to the total length of the Data Field other than that imposed by the programming language.

#### Data Length

The three SSA-NAME3 calls, **ssan3\_get\_keys\_encoded**, **ssan3\_get\_ranges\_encoded** and **ssan3\_match\_encoded** all require the length of the Data field to be supplied.

#### Data Type or Encoding

The three SSA-NAME3 calls, **ssan3\_get\_keys\_encoded**, **ssan3\_get\_ranges\_encoded** and **ssan3\_match\_encoded**, all require the data type or encoding. The encoding could be one of the Unicode encodings or a standard single or double byte character encoding.

The valid values for this field are:

Encoding	Meaning
TEXT	Data is not in a Unicode encoding
UTF-8	Unicode UTF-8 format
UTF-16	Unicode UTF-16 format
UTF-16LE	Unicode UTF-16 format Little Endian
UTF-16BE	Unicode UTF-16 format Big Endian
UTF-32	Unicode UTF-32 format
UCS-2	Same as Unicode UTF-16 format
UCS-4	Unicode UTF-32 format
CP932	Japanese CP932 Codepage (shift-JIS)
CP936	Chinese CP936 Codepage (GBK or Simplified Chinese)
CP949	Korean CP949 Codepage
CP950	Chinese CP950 Codepage (Big5 or Traditional Chinese)
UTF8	All keywords can be specified without the "-"

There are short forms for these values:

Encoding	Meaning
Y	Unicode UTF-8 format
8	Unicode UTF-8 format
6	Unicode UTF-16 format
L	Unicode UTF-16LE format
B	Unicode UTF-16BE format
4	Unicode UCS-4 or UTF-32 format
J	Japanese CP932 codepage (Shift-JIS)
S	Chinese CP936 codepage (Simplified Chinese)
K	Korean CP949 codepage
T	Chinese CP950 codepage (Traditional Chinese)

## Return Value

For those languages which support the returning of a value from a function, all the SSA-NAME3 API calls return a `Long`. If the call succeeded, the value will be zero. Otherwise, the value will be less than zero.

The first operation that the user program should perform after making a call to an SSA-NAME3 API is to check if the returned value is less than 0. If it is, then a catastrophic error has occurred and processing should not continue. If not less than zero, then the call was performed successfully, although a logical error may have been detected. E.g. an unknown keyword may have been found in the Controls parameter. This type of error is indicated by the Response Code field. For this reason, the user program should always check the Response Code field after checking that the returned value is not less than zero. Here is a snippet of C code to demonstrate:

```
rc = ssan3_open (sockh, session_id, sysName,
                population, controls,
                rsp_code, SSA_SI_RSP_SZ,
                n3_msg, SSA_SI_SSA_MSG_SZ);

if (rc < 0) {
    fprintf (stderr, "%s> Error %ld from ssan3_open
\n", prog, rc);
    exit (-17);
}
if (rsp_code[0] != '0') {
    fprintf (stderr, "%s> '%s'\n", prog, n3_msg);
    exit (-18);
}
/* do some more */
```

# ssan3\_connect

An optional function used to connect to an SSA-NAME3 Server through TCP/IP.

## Parameters

Out	Socket Handle	Refer to the section Common Parameters
In	Host	IP Address or Name of remote computer
In	Port	Port Number SSA-NAME3 is Listening on [default is 1665]
Out	Response Code	Refer to the section Common Parameters
Out	Error Message	Refer to the section SSA-NAME3 Error Messages

For z/OS COBOL this function is named `SSACONN`.

# ssan3\_disconnect

An optional function used to disconnect from the SSA-NAME3 Server. This API can be used only if a **ssan3\_connect** API was previously invoked.

## Parameters

In	Socket Handle	Refer to the section Common Parameters
Out	Response Code	Refer to the section Common Parameters
Out	Error Message	Refer to the section SSA-NAME3 Error Messages

For z/OS COBOL this function is named `SSADISC`.

# ssan3\_open

This function opens and initiates an SSA-NAME3 session in preparation for using further API functions. It can also be used to set or override the `SSAPR` and `TIMEOUT` environment variables.

## Parameters

In	Socket Handle	Refer to the Common Parameters section
Out	Session ID	Refer to the Common Parameters section
In	System	Refer to the Common Parameters section

In	Population	Refer to the Common Parameters section
In	Controls	Refer to the Controls section
Out	Response Code	Refer to the Common Parameters section
Out	Error Message	When Response Code is not zero, refer to the SSANAME3 Error Messages section. When <code>Response Code</code> is zero, this field will contain signatures of the core routine and population.

For z/OS COBOL this function is named `SSAOPEN`.

**Note:** The use of the `SSA-NAME3 CJK-SUPPORT` double-byte populations requires a separate license. Ensure you contact the local Informatica Corporation support office for details.

## ssan3\_close

This API closes the SSA-NAME3 session and releases memory. The session is then available for reuse.

### Parameters

In	Socket Handle	Refer to the Common Parameters section
In/Out	Session ID	Refer to the Common Parameters section
In	System	Refer to the Common Parameters section
In	Population	Refer to the Common Parameters section
In	Controls	Refer to the Controls section
Out	Response Code	Refer to the Common Parameters section
Out	Error Message	Refer to the SSA-NAME3 Error Messages section

For z/OS COBOL this function is named `SSACLOSE`.

## ssan3\_get\_keys\_encoded

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA-NAME3 Key table. This is an extended version of the `ssan3_get_keys` function. It has two extra parameters that describe the length and type of the Key Field Data.

## Parameters

In	Socket Handle	Refer to the Common Parameters section
In/Out	Session ID	Refer to the Common Parameters section
In	System	Refer to the Common Parameters section
In	Population	Refer to the Common Parameters section
In	Controls	Refer to the Controls section
Out	Response Code	Refer to the Common Parameters section
Out	Error Message	Refer to the SSA-NAME3 Error Messages section
In	Key Field Data	Refer to the Common Parameters section
In	Key Field Data Size	Refer to the Common Parameters section
In	Key Field Data Type or Encoding	Refer to the Common Parameters section
Out	Keys Count	A number defining the actual number of keys returned for this name or address
Out	Required Keys	An array of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

When using `KEY_SIZE=5`, you will still allocate space for 8 byte keys, however the last 3 bytes of each key is unused.

For z/OS COBOL this function is not yet supported. Use the **ssan3\_get\_keys** function.

### Limitations

**Required Keys:** **ssan3\_get\_keys** will return a maximum of 1024 keys. The `MAX_ENTRIES` Control can be used to specify a lower maximum value.

**Key Size:** 8 byte keys must be used in an application environment that uses the SSA-NAME3 server.

## ssan3\_get\_keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA-NAME3 Key table.

This function has been replaced by **ssan3\_get\_keys\_encoded**.

## Parameters

In	Socket Handle	Refer to the Common Parameters section
In/Out	Session ID	Refer to the Common Parameters section
In	System	Refer to the Common Parameters section
In	Population	Refer to the Common Parameters section
In	Controls	Refer to the Controls section
Out	Response Code	Refer to the Common Parameters section
Out	Error Message	Refer to the SSA-NAME3 Error Messages section
In	Key Field Data	Refer to the Common Parameters section
Out	Keys Count	A number defining the actual number of keys returned for this name or address
Out	Required Keys	An array of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

For z/OS COBOL this function is named `GETKEYS`.

When using `KEY_SIZE=5`, you will still allocate space for 8 byte keys, however the last 3 bytes of each key is unused.

## Limitations

Required Keys: `ssan3_get_keys` will return a maximum of 1024 keys. The `MAX_ENTRIES` Control can be used to specify a lower maximum value.

Key Size: 8 byte keys must be used in an application environment that uses the SSA-NAME3 server.

# ssan3.get\_keys\_raw()

Gets the SSA-NAME3 keys from the SSA-NAME3 key table as a byte array for a name or address.

## Prototype

The following prototype shows the usage of the `ssan3.get_keys_raw()` function:

```
using ssa.ssaname3 ;
public synchronized int ssan3_get_keys_raw (
    int []    session ,    // Integer i/o
    String   system ,    // String in
    String   population , // String in
    String   controls ,  // String in
    String [] rsp ,      // String out
    String [] ssaMsg ,   // String out
    byte[]   data ,      // ByteArray in
    String   dataEncType ,
    int []   count ,     // Integer out
    byte []  keys ,      // ByteArray out
    int []   maxKeys ,   // Integer in
```



```

        int []    keySize    // Integer in
    ) throws SSAPIException , SSASocketException ;

```

## Parameters

The following table lists the parameters of the `ssan3.get_keys_raw()` function:

Input/Output Type	Parameter	Description
In/Out	Session ID	Refer to the Common Parameters section.
In	System	Refer to the Common Parameters section.
In	Population	Refer to the Common Parameters section.
In	Controls	Refer to the Common Parameters section.
Out	Response code	Refer to the Common Parameters section.
Out	Error message	Refer to the SSA-NAME3 Error Messages section.
In	Key field data	Data to be converted to a byte array.
	Encoding	Refer to the Common Parameters section.
Out	Keys count	Number of keys to return for the name or address data.
Out	Keys	Array of keys that the function returns as a byte array.
In	Maximum keys	Maximum number of keys to return. You can use any number from 1024 to 65535.
In	Key size	Size of each key, which can be 5, 8, or 12 bytes.

## RELATED TOPICS:

- [“Common Parameters” on page 23](#)
- [“SSA-NAME3 Error Messages” on page 320](#)

# ssan3\_convert\_keys

An optional function used to convert the SSA-NAME3 8-byte Keys, passed from the SSA-NAME3 Server, into 5-byte binary Keys. You may choose to store 5-byte binary Keys in order to save on disk space.

Note that some databases, programming languages and transport mechanisms will not support binary keys. If you are unsure, please contact Informatica Corporation.

## Parameters

In	Socket Handle	Refer to the Common Parameters section
In	Keys Count	A number defining the actual number of keys to be converted

In	8-byte Keys	An array of 8-byte keys, containing the number of keys defined in Keys Count
In	8-byte Keys Size	Size of the array of 8-byte keys in bytes
Out	5-byte Keys	An array of 5-byte keys, containing the number of keys defined in Keys Count
In	5-byte Keys Size	Size of the array of 5-byte keys
Out	Response Code	Refer to the Common Parameters section
Out	Error Message	Refer to the SSA-NAME3 Error Messages section

For z/OS COBOL this function is named `CONVKEYS`. Refer to the sample code for more information.

## ssan3\_get\_ranges\_encoded

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA-NAME3 Key table. This is an extended version of the **ssan3\_get\_ranges** function. It has two extra parameters that describe the length and type of the Key Field Data.

### Parameters

In	Socket Handle	Refer to the Common Parameters section
In/Out	Session ID	Refer to the Common Parameters section
In	System	Refer to the Common Parameters section
In	Population	Refer to the Common Parameters section
In	Controls	Refer to the <b>ssan3_get_keys_encoded</b> Controls section
Out	Response Code	Refer to the Common Parameters section
Out	Error Message	Refer to the SSA-NAME3 Error Messages section
In	Key Field Data	Refer to the Common Parameters section
In	Key Field Data Size	Refer to the Common Parameters section
In	Key Field Data Type or Encoding	Refer to the Common Parameters section

Out	Ranges Count	A number defining the actual number of key ranges returned for this name or address
Out	Ranges Array	An array of pairs of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address.

Contained in the `Ranges Array` will be a set of "Start" and "End" Key values. These should be used by the calling program to form a set of SQL select statements, or one SQL select statement with a number of `OR`'s.

For example, the `Ranges Array` may contain the following Ranges:

```
Range #1: Y/$$$$$$ Y/$$$$$$/
Range #2: WR$$$$$$ WR$$$$$/
```

In which case the calling program will select records from the SSA-NAME3 Key Table where SSANAME3 Key values are:

```
GREATER THAN OR EQUAL TO "Y/$$$$$$" AND LESS THAN OR EQUAL TO "Y/$$$$$/"
OR
GREATER THAN OR EQUAL TO "WR$$$$$" AND LESS THAN OR EQUAL TO "WR$$$$/".
```

When using `KEY_SIZE=5`, you will still allocate space for 8 byte key ranges (that is, 16 bytes), however the last 6 bytes of each range pair is unused (that is, only the first 10 bytes are used).

For z/OS COBOL this function is not yet supported. Use the `ssan3_get_ranges` function.

### Limitations

Required Ranges: `ssan3_get_ranges_encoded` will return a maximum of 1024 ranges. The `MAX_ENTRIES` Control can be used to specify a lower maximum value.

Key Size: 8 byte keys must be used in an application environment that uses the SSA-NAME3 server.

## ssan3\_get\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA-NAME3 Key table.

This function has been replaced by `ssan3_get_ranges_encoded`.

### Parameters

For z/OS COBOL this function is named `GETRNGES`.

Contained in the `Ranges Array` will be a set of "Start" and "End" Key values. These should be used by the calling program to form a set of SQL select statements, or one SQL select statement with a number of `OR`'s.

For example, the `Ranges Array` may contain the following Ranges:

```
Range #1: Y/$$$$$$ Y/$$$$$$/
Range #2: WR$$$$$$ WR$$$$$/
```

In which case the calling program will select records from the SSA-NAME3 Key Table where SSANAME3 Key values are:

```
GREATER THAN OR EQUAL TO "Y/$$$$$$" AND LESS THAN OR EQUAL TO "Y/$$$$$/"
```

OR  
 GREATER THAN OR EQUAL TO "WR\$\$\$\$\$" AND LESS THAN OR EQUAL TO "WR\$\$\$\$\$/"

When using `KEY_SIZE=5`, you will still allocate space for 8 byte key ranges (that is, 16 bytes), however the last 6 bytes of each range pair is unused (that is, only the first 10 bytes are used).

### Limitations

**Required Ranges:** `ssan3_get_ranges` will return a maximum of 1024 ranges. The `MAX_ENTRIES` Control can be used to specify a lower maximum value.

**Key Size:** 8 byte keys must be used in an application environment that uses the SSA-NAME3 server.

## ssan3.get\_ranges\_raw()

Gets the SSA-NAME3 key ranges as a byte array for a name or address. An application program uses the key ranges to retrieve records from the SSA-NAME3 key table.

### Sample

The following prototype shows the usage of the `ssan3.get_ranges_raw()` function:

```
using ssa.ssaname3 ;
public synchronized int ssan3_get_ranges_raw (
    int []    session ,    // Integer i/o
    String    system ,    // String in
    String    population , // String in
    String    controls ,  // String in
    String [] rsp ,       // String out
    String [] ssaMsg ,    // String out
    byte []   data ,      // ByteArray in
    String    dataEncType ,
    int []    count ,     // Integer out
    byte []   stab ,      // ByteArray out
    int []    maxRanges , // Integer in
    int []    RangeSize  // Integer in
) throws SSAAPIException , SSASocketException ;
```

### Parameters

The following table lists the parameters of the `ssan3.get_ranges_raw()` function:

Input/Output Type	Parameter	Description
In/Out	Session ID	Refer to the Common Parameters section.
In	System	Refer to the Common Parameters section.
In	Population	Refer to the Common Parameters section.
In	Controls	Refer to the Common Parameters section.
Out	Response code	Refer to the Common Parameters section.
Out	Error message	Refer to the SSA-NAME3 Error Messages section.
In	Key field data	Data to be converted to a byte array.

Input/Output Type	Parameter	Description
	Encoding	Refer to the Common Parameters section.
Out	Keys count	Number of keys to return for the name or address data.
Out	Keys	Array of key pairs that the function returns as a byte array.
In	Maximum ranges	Maximum number of key ranges to return. Use any number from 1024 to 65535.
In	Range size	Size of each key range, which can be 10, 16, or 24 bytes. Based on the key size, use the following sizes for a key range: - 10 for 5-byte keys. - 16 for 8-byte keys. - 24 for 12-byte keys.

#### RELATED TOPICS:

- [“Common Parameters” on page 23](#)
- [“SSA-NAME3 Error Messages” on page 320](#)

## ssan3\_convert\_ranges

An optional function used to convert the SSA-NAME3 8-byte Key Ranges, passed from the SSANAME3 Server, into 5-byte binary Key Ranges. You will use this function if you have stored SSANAME3 5-byte binary Keys on your database.

#### Parameters

For z/OS COBOL this function is named `CONVRNGS`.

## ssan3\_match\_encoded

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges\_encoded** call, **ssan3\_match\_encoded** is called to further qualify the candidate records.

#### Parameters

For z/OS COBOL this function is not yet supported. Use the **ssan3\_match** function.

# ssan3\_match

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

This function has been replaced by **ssan3\_match\_encoded**.

## Parameters

For z/OS COBOL this function is named `SSAMATCH`.

# ssan3\_get\_word\_stack

Retrieves the word stack containing an array of stabilized words for an input string. The **ssan3\_get\_word\_stack** function retrieves the word stack for multiple strings.

## Parameters

In	Socket Handle	Refer to the Common Parameters section.
In/Out	Session ID	Refer to the Common Parameters section.
In	System	Refer to the Common Parameters section.
In	Population	Refer to the Common Parameters section.
In	Controls	Refer to the Controls section.
Out	Response Code	Refer to the Common Parameters section.
Out	Error Message	Refer to the SSA-NAME3 Error Messages section.
In	Key Field Data	Refer to the Common Parameters section.
Out	Word Stack	<p>A word stack containing an array of stabilized words for the input string.</p> <p>The <code>ssan3_get_word_stack</code> call returns the word stack in the following format:</p> <pre>Content: Word / Type / Original Type / Category / Original Initial / Stabilized word Offset:  0   24   25           26       28           32 Length: 24   1   1           2         1           24</pre> <p>For example, the <code>ssan3_get_word_stack</code> call returns the following output:</p> <pre>0 'MICHAEL Y NKM MACAL ' 1 'TAYLOR MY T TALAR  '</pre>

## ssan3\_match\_ext

Returns a score, match decision, and match report for the search and file records. After the candidate records are identified based on the ranges that the **ssan3\_get\_ranges\_ext** call returns, the **ssan3\_match\_ext** call qualifies the candidate records.

## ssan3\_match\_rescore

Returns the score and the match decision based on the revised field weights.

## ssan3\_info

Returns information about the selected system and population.

### Parameters

For z/OS COBOL this function is named `SSAINFO`.

## ssan3\_info\_ext

Returns information about the selected system and population for the following **ssan3\_info** control items:

- map\_profile
- map\_population
- map\_data
- map\_field

For more information about `ssan3_info` control items, see [“ssan3\\_info Controls” on page 310](#).

## CHAPTER 4

# Language Specific Guidelines

This chapter provides tips and techniques when programming using specific programming languages. It should be read in conjunction with the sample code generated from the Developer's Workbench as well as the information provided in the Functions chapter.

## Data Types

The section discusses the SSA-NAME3 data types used in this document. The *Language Specific Bindings* section describes the mapping between the SSA-NAME3 data types and the native data types used for specific programming languages.

### Strings

The `String` data-type is a variable length piece of memory, terminated with a NUL character (0x00).

When using a `String` as an input parameter there is no need to explicitly tell SSA-NAME3 how long it is because SSA-NAME3 can detect its length.

When the API returns a `String` as an output parameter, the caller must allocate memory for it and in some programming languages, tell SSA-NAME3 how long it is. For example, in C it is not possible to detect how long a piece of memory is, so the caller must pass two parameters:

- the address of the memory, and
- its length.

SSA-NAME3 uses this information to prevent overwriting unallocated memory (which would result in GPF or core dump).

### String Arrays

A `StringArray` is an array of Strings. It consists of an array of pointers, with each pointer pointing to a String. A String Array is usually passed through the API using three parameters:

- address of the array of pointers,
- number of pointers in the array, and
- the length of each String.

Note that since there is provision for only one length value, all Strings must be the same length.



## Blocks

A `Block` data-type is a fixed length piece of memory. It is not NUL terminated. If a value is not long enough to fill the entire Block, the Block should be padded with spaces on the right. Since SSANAME3 cannot detect how long the memory is, Blocks are usually passed through the API using two parameters:

- a pointer to the memory, and
- the length of the Block

## Block Arrays

A `BlockArray` is an array of Blocks. It consists of an array of pointers, with each pointer pointing to a Block. A Block Array is usually passed through the API using three parameters:

- address of the array of pointers,
- number of pointers in the array, and
- the length of each Block.

**Note:** Since there is provision for only one length value, all Blocks must be the same length.

## Nulls and NULs

Some programming languages permit the use of Null pointers (such as C). Null pointers must never be passed as arguments to any API functions. If you do not wish to provide a value for an argument, use a NUL terminated (0x00) string instead (a zero-length, empty string).

# Language Specific Bindings

Sample programs can be generated from the Developer's Workbench in the following programming languages:

- C
- C#
- C++
- Cobol (CICS)
- Cobol (z/OS)
- Java
- Microsoft SQL Server
- Perl
- Visual Basic 6
- Visual Basic .NET

# Calling from C

## ssan3\_addr\_get\_cass\_field

Use this function to retrieve cass specific address fields.

### Prototype

```
long ssan3_addr_get_class_field (  
    long sockh ,           // Long in  
    long *session ,       // Long io  
    char *rsp ,           // String out  
    long rsp_size ,  
    char *ssaMsg ,        // String out  
    long ssaMsg_size ,  
    long suggest_idx ,    // Long in  
    long field_idx ,      // Long in  
    char *field_value ,   // Block out  
    long field_value_size  
);
```

### Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a cass field
field_idx	Specifies a cass field within the nth suggestion
field_value	The cass field value

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_cass\_field\_cnt

Use this function to determine the max number of cass address fields

### Prototype

```
long ssan3_addr_get_cass_field_cnt (  
    long sockh ,           // Long in  
    long *session ,       // Long io  
    char *rsp ,           // String out  
    long rsp_size ,  
    char *ssaMsg ,        // String out  
    long ssaMsg_size ,  
    long *count           // Long out  
);
```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of cass address fields

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_cass\_field\_info

Use this function to retrieve information about a suggestion.

### Prototype

```
long ssan3_addr_get_cass_field_info (
    long sockh ,           // Long in
    long *session ,       // Long io
    char *rsp ,           // String out
    long rsp_size ,       // String out
    char *ssaMsg ,        // String out
    long ssaMsg_size ,    // String out
    long suggest_idx ,    // Long in
    long *field_length ,  // LongArray out
    long field_length_num
) ;
```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each cass address field

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_del\_lines

Use this function to retrieve delivery address line information, This function is deprecated, use **addr\_get\_del\_lines\_ext** instead

## Prototype

```
long ssan3_addr_get_del_lines (
    long sockh ,           // Long in
    long *session ,       // Long io
    char *rsp ,           // String out
    long rsp_size ,
    char *ssaMsg ,        // String out
    long ssaMsg_size ,
    long suggest_idx ,    // Long in
    char *del_line 1 ,    // Block out
    long del_line 1_size ,
    char *del_line 2 ,    // Block out
    long del_line 2_size ,
    char *del_line 3 ,    // Block out
    long del_line 3_size ,
    char *del_line 4 ,    // Block out
    long del_line 4_size ,
    char *del_line 5 ,    // Block out
    long del_line 5_size ,
    char *del_line 6 ,    // Block out
    long del_line 6_size
) ;
```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameter section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_line1	Delivery address line 1 output string
del_line2	Delivery address line 2 output string
del_line3	Delivery address line 3 output string
del_line4	Delivery address line 4 output string
del_line5	Delivery address line 5 output string
del_line6	Delivery address line 6 output string

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_del\_lines\_ext

Use this function to retrieve delivery address line information.

## Prototype

```
long ssan3_addr_get_del_lines_ext (
    long sockh ,           // Long in
```

```

long    *session ,          // Long io
char    *rsp ,             // String out
long    rsp_size ,
char    *ssaMsg ,         // String out
long    ssaMsg_size ,
long    suggest_idx ,     // Long in
long    del_case ,        // Long in
char    *del_line 1 ,     // Block out
long    del_line 1_size ,
char    *del_line 2 ,     // Block out
long    del_line 2_size ,
char    *del_line 3 ,     // Block out
long    del_line 3_size ,
char    *del_line 4 ,     // Block out
long    del_line 4_size ,
char    *del_line 5 ,     // Block out
long    del_line 5_size ,
char    *del_line 6 ,     // Block out
long    del_line 6_size
) ;

```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_case	Specifies delivery address line case option value. The allowed values are 0 = Unchanged, 1= Upper case, 2 = Lower case and 3 = Mixed case.
del_line1	Delivery address line 1 output string
del_line2	Delivery address line 2 output string
del_line3	Delivery address line 3 output string
del_line4	delivery address line 4 output string
del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field

Use this function to retrieve validated address fields.

### Prototype

```

long ssan3_addr_get_field (
sockh , // Long in

```

```

    session ,                long *          //          Long io
    rsp ,                    char *          //          String out
                             long            rsp_size ,
    ssaMsg ,                 char *          //          String out
                             long            ssaMsg_size ,
    suggest_idx ,           long            Long in
    field_idx ,             long            Long in
    Block out               char *          field_value , //
                             long            field_value_size ,
                             long *         field_val_status , // Long out
                             long *         field_val_mods //   Long out
);

```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a field
field_value	The field value
field_idx	Specifies a field within the nth suggestion
field_val_status	Specifies how this field matched the validation data
field_val_mods	Specifies how this field was modified by validation data

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_count

Use this function to determine the max number of address fields.

### Prototype

```

long ssan3_addr_get_field_count (
    long sockh , // Long in
    long * session , // Long io
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    long * count // Long out
);

```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of address fields

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_ext

Use this function to retrieve all getter fields.

### Prototype

```

long ssan3_addr_get_field_ext (
    sockh ,
    char *   rsp ,
    long   rsp_size ,
    char *   ssaMsg ,
    suggest_idx ,
    field_operation ,
    long   field_item_line ,
    char *   field_type ,
    long   field_value_size
) ;
// Long in
// String out
// String out
// ssaMsg_size ,
// Long in
// Long in
// Long in
// String in
// String in
// Block out

```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get fields
field_operation	Field operation Option 0 for AddressElements Option 1 for AddressLines Option 2 for AddressComplete Option 3 for EnrichmentData Option 4 for ResultDataParameter Option 5 for EnrichmentDataStatus
field_name	Refer AD Result.dtd for field names

field_item_line	Represent field line number or field item number
field_type	Refer AD Result.dtd for field attribute Type
field_value	Cleansed field output

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_idx

Use this function to retrieve individual address fields.

### Prototype

```

long ssan3_addr_get_field_idx (
    sockh ,                               long           // Long in
    session ,                             long *         // Long io
    rsp ,                                  char *         // String out
    ssaMsg ,                               long           rsp_size ,
    suggest_idx ,                          long           ssaMsg_size ,
    field_idx ,                             long           // Long in
    field_value ,                          long           // Long in
    ) ;                                     char *         Block out
                                           long           field_value_size

```

### Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a field. On successful parse, use 0 for ASM for AD version 4, 1 for ASM for AD version 5
field_idx	Specifies a field within the nth suggestion
field_value	The field value

### Return Code

negative for error, 0 for success



## ssan3\_addr\_get\_field\_info\_ext

Use this function to retrieve information about a suggestion

### Prototype

```
long ssan3_addr_get_field_info_ext (
    long sockh , // Long in
    * session , // Long io
    * rsp , // char //
    String out
    long rsp_size ,
    * ssaMsg , // char // String out
    long ssaMsg_size ,
    long suggest_idx , // Long in
    long * field_length , // LongArray out
    long field_length_num ,
    char * addr_label_encoded , // Block out
    long addr_label_encoded_size ,
    char * addr_label_charset , // String out
    long addr_label_charset_size ,
    long * score // Long out
) ;
```

### Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each address field
addr_label_encoded	The returned label
addr_label_charset	The character set used in the address label
score	The returned label's score

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_len

Use this function to determine the max field length

## Prototype

```
long ssan3_addr_get_field_len (
    sockh ,                // Long in
    session ,              // Long io
    long    rsp_size ,    char *    rsp ,        // String out
    long    ssaMsg_size , char *    ssaMsg ,    // String out
    long *  max_len      // Long out
) ;
```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max address field length in bytes

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_line\_len

### Prototype

```
long ssan3_addr_get_line_len (
    sockh ,                // Long in
    session ,              // Long io
    long    rsp_size ,    char *    rsp ,        // String out
    ssaMsg ,              // String out
    long    ssaMsg_size , char *
    long *  max_len      // Long out
) ;
```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max line length in bytes

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_option

Use this function to set optional parameters.

### Prototype

```
long ssan3_addr_get_option (
    sockh , // long
    session , // Long in
    char * // Long io
    rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    char * param, // String in
    char * value , // String out
    long value_size
) ;
```

### Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to get.
value	Returns the value for the option.

## Return Code

negative for error, 0 for success

## ssan3\_addr\_init

Use this function to initialize the Address Standardization interface.

### Prototype

```
long ssan3_addr_init (
    long sockh , // Long in
    long * session , // Long io
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    long max_memory // Long in
) ;
```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_memory	This field specifies the maximum amount of memory (MB) to be used by the Address Standarization engine

## Return Code

negative for error, 0 for success

## ssan3\_addr\_parse

Use this function to parse an address.

### Prototype

```
long ssan3_addr_parse (
    long sockh , // Long in
    long * session , // Long io
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    long * field_length , // LongArray out
    long field_length_num
) ;
```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_length	An array containing the length of each parsed field

## Return Code

negative for error, 0 for success

## ssan3\_addr\_preload\_country

Use this function to preload country database.

### Prototype

```
long ssan3_addr_preload_country (
```

```

    long sockh , // Long in
    long * session , // Long io
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    char * preload_type , // String in
    char * preload_country , // String in
    char * val_mode // String in
);

```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
preload_type	type of preload to perform
preload_country	country database to be preloaded
val_mode	This field specified validation mode

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_attrib

Use this function to specify the character set of the data and a default country.

## Prototype

```

long ssan3_addr_set_attrib (
    long sockh , // Long in
    long * session , // Long io
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    char * char_set , // String in
    char * default_country // String in
);

```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

char_set	The name of the character set used to encode the input and output.
default_country	The default country used for validation when parsing cannot detect a country name.

### Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_del\_lines

Use this function to set delivery address line information.

### Prototype

```

long ssan3_addr_set_del_lines (
    long sockh ,           // Long in
    long * session ,      // Long io
    char * rsp ,          // String out
    long rsp_size ,
    char * ssaMsg ,       // String out
    long ssaMsg_size ,
    char * del_line1 ,    // Block in
    long del_line1_size ,
    char * del_line2 ,    // Block in
    long del_line2_size ,
    char * del_line3 ,    // Block in
    long del_line3_size ,
    char * del_line4 ,    // Block in
    long del_line4_size ,
    char * del_line5 ,    // Block in
    long del_line5_size ,
    char * del_line6 ,    // Block in
    long del_line6_size
) ;

```

### Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
del_line1	delivery address line 1 input string
del_line2	delivery address line 2 input string
del_line3	delivery address line 3 input string
del_line4	delivery address line 4 input string
del_line5	delivery address line 5 input string
del_line6	delivery address line 6 input string

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_field\_case

Use this function to set individual input fields case option

### Prototype

```
long ssan3_addr_set_field_case (
    long sockh , // Long in
    Long io
    *
    char * session , // Long io
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    long field_idx , // Long in
    long field_case // Long in
) ;
```

### Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_case	Specifies output field case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case.

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_field\_idx

Use this function to set individual input fields by idx

### Prototype

```
long ssan3_addr_set_field_idx (
    long sockh , // Long in
    Long io
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    long field_idx , // Long in
    char * field_value , // Block in
    long session , // Long in
) ;
```

```

        long    field_value_size
    ) ;

```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_value	Specifies a value for the nth field

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_field\_name

Use this function to set individual input fields by name

### Prototype

```

long ssan3_addr_set_field_name (
long    sockh ,           // Long in
long * session ,         // Long io
char *  rsp ,            // String out
long   rsp_size ,
char *  ssaMsg ,         // String out
long   ssaMsg_size ,
char *  field_name ,    // String in
char *  field_value ,   // Block in
long   field_value_size
) ;

```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_name	Specifies the name of the field to set
field_value	Specifies a value for the field

## Return Code

negative for error, 0 for success



## ssan3\_addr\_set\_lines

Use this function to provide an address to parse or validate

### Prototype

```
long ssan3_addr_set_lines (  
    long sockh ,           // Long in  
    long * session ,      // Long io  
    char * rsp ,          // String out  
    long rsp_size ,  
    char * ssaMsg ,       // String out  
    long ssaMsg_size ,  
    char * line_1 ,       // Block in  
    long line_1_size ,  
    char * line_2 ,       // Block in  
    long line_2_size ,  
    char * line_3 ,       // Block in  
    long line_3_size ,  
    char * line_4 ,       // Block in  
    long line_4_size ,  
    char * line_5 ,       // Block in  
    long line_5_size ,  
    char * line_6 ,       // Block in  
    long line_6_size ,  
    char * line_7 ,       // Block in  
    long line_7_size ,  
    char * line_8 ,       // Block in  
    long line_8_size ,  
    char * line_9 ,       // Block in  
    long line_9_size ,  
    char * line_10 ,      // Block in  
    long line_10_size  
);
```

### Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
line_1	The first line of the address
line_2	The second line of the address
line_3	The third line of the address
line_4	The fourth line of the address
line_5	The fifth line of the address
line_6	The sixth line of the address
line_7	The seventh line of the address
line_8	The eighth line of the address

line_9	The ninth line of the address
line_10	The tenth line of the address

### Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_option

Use this function to set optional parameters

### Prototype

```

long ssan3_addr_set_option (
long sockh ,          // Long in
long * session ,     // Long io
char * rsp ,         // String out
long  rsp_size ,
char * ssaMsg ,     // String out
long  ssaMsg_size ,
char * param,       // String in
char * value        // String in
) ;

```

### Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to set.
value	This field specifies a value for the option.

### Return Code

negative for error, 0 for success

## ssan3\_addr\_validate

Use this function to validate an address.

### Prototype

```

long ssan3_addr_validate (
long sockh ,          // Long in
long * session ,     // Long io
char * rsp ,         // String out
long  rsp_size ,
char * ssaMsg ,     // String out
long  ssaMsg_size ,
long * status ,     // Long out

```

```

    long * n_suggest // Long out
);

```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
status	The status returned by the validation process
n_suggest	The number of suggestions generated by validation

## Return Code

negative for error, 0 for success

## ssan3\_close

Closes the SSA session and deallocates memory. Similarly to **ssa\_open**, it is optional.

## Prototype

```

long ssan3_close (
    long sockh , // Long in
    long * session , // Long io
    char * system , // String in
    char * population , // String in
    char * controls , // String in
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size
);

```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## ssan3\_connect

Initiates a socket.

### Prototype

```
long    ssan3_connect (
char *  host ,          // String in
long    port ,         // Long in
long *  sockh          // Long out
) ;
```

### Parameters

host	This is the host to connect to.
port	This is the port to connect to.
sockh	This is a socket handle.

### Return Code

negative for error, 0 for success

## ssan3\_convert\_keys

Used to convert 8 byte keys (from server) to 5 byte keys (for client).

### Prototype

```
long    ssan3_convert_keys (
long    sockh ,        // Long in
char ** keys8 ,        // StringArray in
long    keys8_num ,
char ** keys5 ,        // BlockArray out
long    keys5_num ,
long    keys5_size ,
char *  rsp ,          // String out
long    rsp_size ,
char *  ssaMsg ,      // String out
long    ssaMsg_size
) ;
```

### Parameters

sockh	This is the socket to use for the call
keys8	block of 8 byte keys to be converted
keys5	block of 5 byte keys (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## ssan3\_convert\_ranges

Used to convert 8 byte ranges (from server) to 5 byte ranges (for client).

### Prototype

```
long ssan3_convert_ranges (  
    long sockh , // Long in  
    char ** ranges8 , // StringArray in  
    long ranges8_num ,  
    char ** ranges5 , // BlockArray out  
    long ranges5_num ,  
    long ranges5_size ,  
    char * rsp , // String out  
    long rsp_size ,  
    char * ssaMsg , // String out  
    long ssaMsg_size  
);
```

### Parameters

sockh	This is the socket to use for the call
ranges8	block of 8 byte ranges to be converted
ranges5	block of 5 byte ranges (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## ssan3\_disconnect

Releases resources allocated to a socket.

### Prototype

```
long ssan3_disconnect (  
    long sockh // Long in  
);
```

### Parameters

sockh	This is the socket to use for the call
-------	--

### Return Code

negative for error, 0 for success

## ssan3\_errors\_get\_all

Get the Server side error messages from the last API function that failed. This function should be called repeatedly until it returns 1, meaning all messages have been retrieved.

**Note:** If a communication (socket) error occurred, this function will also fail. Refer to the *OPERATIONS MANUAL, Error Log* section for information on interpreting the Error Log.

## Prototype

```
long ssan3_errors_get_all (
long sockh , // Long in
char * msg , // String out
long msg_size
) ;
```

## Parameters

sockh	This is the socket to use for the call
msg	This is an error message.

## Return Code

negative for error, 0 for success

# ssan3\_get\_keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

## Prototype

```
long ssan3_get_keys (
long sockh , // Long in
long * session , // Long io
char * system , // String in
char * population , // String in
char * controls , // String in
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
char * data , // String in
long * count , // Long out
char ** keys , // StringArray out
long keys_size
) ;
```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or address

### Return Code

negative for error, 0 for success

## ssan3\_get\_keys\_encoded

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```

long ssan3_get_keys_encoded (
    long sockh , // Long in
    long * session , // Long io
    char * system , // String in
    char * population , // String in
    char * controls , // String in
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    char * data , // EncodedString in
    long dataSize ,
    char * dataEncType ,
    long * count , // Long out
    char ** keys , // StringArray out
    long keys_size
) ;

```

### Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address keys An array of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or address

## Return Code

negative for error, 0 for success

## ssan3\_get\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```
long ssan3_get_ranges (
long sockh , // Long in
long * session , // Long io
char * system , // String in
char * population , // String in
char * controls , // String in
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
char * data , // String in
long * count , // Long out
char ** stab , // StringArray out
long stab_size
) ;
```

### Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or address

## Return Code

negative for error, 0 for success

## ssan3\_get\_ranges\_encoded

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.



## Prototype

```
long ssan3_get_ranges_encoded (
long sockh , // Long in
long * session , // Long io
char * system , // String in
char * population , // String in
char * controls , // String in
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
char * data , // EncodedString in
long dataSize ,
char * dataEncType ,
long * count , // Long out
char ** stab , // StringArray out
long stab_size
) ;
```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or address

## Return Code

negative for error, 0 for success

## ssan3\_info

Returns Information about the selected system and population.

## Prototype

```
long ssan3_info (
long sockh , // Long in
long * session , // Long io
char * system , // String in
char * population , // String in
char * controls , // String in
char * rsp , // String out
long rsp_size ,
```

```

char *   ssaMsg , //      String out
long    ssaMsg_size ,
long *  count , //      Long out
char ** info , //      StringArray out
long    info_size
) ;

```

## Parameters

sockh	Socket to use for the call.
session	Refer to the Common Parameters section.
system	Refer to the Common Parameters section.
population	Refer to the Common Parameters section.
controls	Refer to the Controls section.
rsp	Refer to the Common Parameters section.
ssaMsg	Refer to the SSA-NAME3 Error Messages section.
count	A number defining the number of rows in the Items Array info.
info	An array of 1024 byte rows, containing the data returned for the specific <b>ssan3_info</b> call.

## Return Code

negative for error, 0 for success

## ssan3\_info\_ext

Returns information about the selected system and population for the following **ssan3\_info** control items:

- map\_profile
- map\_population
- map\_data
- map\_field

## Prototype

```

long ssan3_info (
long    sockh , // Long in
long *  session , // Long io
char *  system_name , // String in
char *  population , // String in
char *  controls , // String in
char *  rsp , // String out
long    rsp_size ,
char *  ssaMsg , // String out
long    ssaMsg_size ,
long *  in_count , // Long in
char ** in_info , // StringArray in
long *  in_info_num ,
long *  count , // Long out
char ** info , // StringArray out
long    info_size
) ;

```

## Parameters

sockh	Socket to use for the call.
session	Refer to the Common Parameters section.
system	Refer to the Common Parameters section.
population	Refer to the Common Parameters section.
controls	Refer to the Controls section.
rsp	Refer to the Common Parameters section.
ssaMsg	Refer to the SSA-NAME3 Error Messages section.
in_count	Number of records that the in_info parameter returns.
in_info	Array of records for the following ssan3_info control items: <ul style="list-style-type: none"><li>- map_profile</li><li>- map_population</li><li>- map_data</li><li>- map_field</li></ul>
count	A number defining the number of rows in the Items Array.
info	An array of 1024 byte rows, containing the data returned for the specific <b>ssan3_info</b> call.

## Return Code

Negative for error and 0 for success.

## ssan3\_keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

## Prototype

```
long ssan3_keys (
long   sockh ,           // Long in
long * session ,        // Long io
char * system ,         // String in
char * population ,    // String in
char * controls ,      // String in
char * rsp ,           // String out
long   rsp_size ,
char * ssaMsg ,        // String out
long   ssaMsg_size ,
char * data ,          // Block in
long   data_size ,
char * encoding ,     // String in
long * count ,        // Long out
char ** keys ,        // BlockArray out
long   keys_num,
long   keys_size
) ;
```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field
count	A number defining the actual number of keys returned for this name or address
keys	An array of keys returned for this name or address

## Return Code

negative for error, 0 for success

## ssan3\_match

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

## Prototype

```
long ssan3_match (
long sockh ,           // Long in
long * session ,      // Long io
char * system ,       // String in
char * population ,   // String in
char * controls ,     // String in
char * rsp ,          // String out
long  rsp_size ,      //
char * ssaMsg ,       // String out
long  ssaMsg_size ,   //
char * search ,       // String in
char * file ,         // String in
char * score ,        // String out
long  score_size ,    //
char * decision ,     // String out
long  decision_size
) ;
```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	refer to CommonParameters section.
file	refer to CommonParameters section.
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

## Return Code

negative for error, 0 for success

## ssan3\_match\_encoded

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

## Prototype

```
long ssan3_match_encoded (
    long sockh , // Long in
    long sockh , // Long in
    long * session , // Long io
    char * system , // String in
    char * population , // String in
    char * controls , // String in
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    char * search , // EncodedString in
    long searchSize ,
    char * searchEncType ,
    char * file , // EncodedString in
    long fileSize ,
    char * fileEncType ,
    char * score , // String out
    long score_size ,
    char * decision , // String out
    long decision_size
);
```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section.
file	Refer to the Common Parameters section.
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

## Return Code

negative for error, 0 for success

## ssan3\_match\_ext

Returns a score, match decision, and match report for the search and file records. After the candidate records are identified based on the ranges that the **ssan3\_get\_ranges\_ext** call returns, the **ssan3\_match\_ext** call qualifies the candidate records.

## Prototype

```
long ssan3_match_ext (
long   sockh ,           // Long in
long * session ,        // Long io
char * system_name ,    // String in
char * population ,     // String in
char * controls ,       // String in
char * rsp ,            // String out
long   rsp_size ,
char * ssaMsg ,         // String out
long   ssaMsg_size ,
char * search ,         // String in
long   s_len ,          // Long in
char * s_encoding ,     // String in
char * file ,           // String in
long   f_len ,          // Long in
char * f_encoding ,     // String in
char * score ,          // String out
long   score_size ,
char * decision ,       // String out
long   decision_size
long * count ,          // Long out
char ** explain_data ,  // StringArray out
```

```

    long * explain_data_size
) ;

```

## Parameters

sockh	Socket to use for the call.
session	Refer to the Common Parameters section.
system	Refer to the Common Parameters section.
population	Refer to the Common Parameters section.
controls	Refer to the Controls section.
rsp	Refer to the Common Parameters section.
ssaMsg	Refer to the SSA-NAME3 Error Messages section.
search	Refer to the Common Parameters section.
file	Refer to the Common Parameters section.
score	A 3-byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, match purpose and match level.
decision	'A' meaning 'Accept'. The score is greater than or equal to the accept limit for the specified match purpose and match level. 'U' meaning 'Undecided'. The score is between the accept limit and reject limit for the specified match purpose and match level. 'R' meaning 'Reject'. The score is less than or equal to the reject limit for the specified match purpose and match level.
explain_data	Returns the match report.

## Return Code

Negative for error and 0 for success.

## ssan3\_match\_rescore

Returns the score and match decision based on the revised field weights.

### Prototype

```

long ssan3_match_rescore (
long    sockh ,           // Long in
long *  session ,        // Long io
char *  system_name ,    // String in
char *  population ,     // String in
char *  controls ,       // String in
char *  rsp ,            // String out
long    rsp_size ,
char *  ssaMsg ,         // String out
long    ssaMsg_size ,
long *  count ,          // Long in
char ** explain_data ,   // StringArray in
long *  explain_data_num ,
char *  score ,          // String out
long    score_size ,

```

```

char * decision ,          // String out
long  decision_size
) ;

```

## Parameters

sockh	Socket to use for the call.
session	Refer to the Common Parameters section.
system	Refer to the Common Parameters section.
population	Refer to the Common Parameters section.
controls	Refer to the Controls section.
rsp	Refer to the Common Parameters section.
ssaMsg	Refer to the SSA-NAME3 Error Messages section.
explain_data	Returns the match report.
score	A 3-byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, match purpose and match level.
decision	'A' meaning 'Accept'. The score is greater than or equal to the accept limit for the specified match purpose and match level. 'U' meaning 'Undecided'. The score is between the accept limit and reject limit for the specified match purpose and match level. 'R' meaning 'Reject'. The score is less than or equal to the reject limit for the specified match purpose and match level.

## Return Code

Negative for error and 0 for success.

## ssan3\_open

An optional function, but its use will improve performance. It opens and initiates an SSA-NAME3 session in preparation.

## Prototype

```

long ssan3_open (
long  sockh ,          // Long in
long * session ,      // Long io
char * system ,       // String in
char * population ,  // String in
char * controls ,     // String in
char * rsp ,          // String out
long  rsp_size ,
char * ssaMsg ,      // String out
long  ssaMsg_size
) ;

```



## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## ssan3\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

## Prototype

```
long ssan3_ranges (
    long sockh , // Long in
    long * session , // Long io
    char * system , // String in
    char * population , // String in
    char * controls , // String in
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    char * data , // Block in
    long data_size ,
    char * encoding , // String in
    long * count , // Long out
    char ** stab , // BlockArray out
    long stab_num ,
    long stab_size
) ;
```

## Parameters

sockh	This is the socket to use for the call
session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section

rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs returned for this name or address

### Return Code

negative for error, 0 for success

## Calling from C#

### Compilation and linking

C# programs should call the SSA-NAME3 DLL as shown in the sample program.

- To uninstall the dll, run the command, `gacutil /u ssan3cs`  
You don't need to do this unless you are installing a new version.
- To install the dll, run the command, `gacutil /i c:\Informatica\IR\bin\ssan3cs.dll`  
You can also achieve this by dragging the dll into `<WindowsDir>\ Assembly`

- Compile:

```
csc /reference:%SSABIN%\ssan3cs.dll sample.cs
```

- Execute:

```
sample
```

### Coding the `ssan3_close` Call outside of the `Finalize()` Method

Calling the `ssan3_close` method from a `Finalize()` method is not recommended. This is because it delays the call to `ssan3_close` until the runtime runs garbage collection on the released object, which can be any time after the object has been released by the program.

Instead, it is recommended that the user implement a public cleanup or close method that performs the necessary housekeeping, including a call to `ssan3_close` to release open sessions prior to releasing the object.

### Exceptions

All methods throw exceptions of type `SSAN3Exception`.

## Common Parameters

Common parameters are accessible as class members.

## addr\_get\_cass\_field

Use this function to retrieve cass specific address fields.

### Prototype

```
using ssa ;  
  
public byte [ ] addr_get_cass_field (  
    int suggest_idx , // Long in  
    int field_idx // Long in  
);
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a cass field
field_idx	Specifies a cass field within the nth suggestion
field_value	The cass field value

### Return Code

negative for error, 0 for success

## addr\_get\_cass\_field\_cnt

Use this function to determine the max number of cass address fields.

### Prototype

```
using ssa ;  
  
public int addr_get_cass_field_cnt ( ) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of cass address fields

### Return Code

negative for error, 0 for success

## addr\_get\_cass\_field\_info

Use this function to retrieve information about a suggestion.

### Prototype

```
using ssa ;  
  
public int [ ] addr_get_cass_field_info (  
    int suggest_idx // Long in  
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each cass address field

### Return Code

negative for error, 0 for success

## addr\_get\_del\_lines

Use this function to retrieve delivery address line information. This function is deprecated, use **addr\_get\_del\_lines\_ext** instead.

### Prototype

```
using ssa ;  
public structaddr_get_del_lines_structaddr_get_del_lines (  
    int suggest_idx // Long in  
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string
del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string

del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

### Return Code

negative for error, 0 for success

## addr\_get\_del\_lines\_ext

Use this function to retrieve delivery address line information.

### Prototype

```
using ssa ;
public struct addr_get_del_lines_ext_struct addr_get_del_lines_ext (
    int          suggest_idx , // Long in
    int          del_case     // Long in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_case	Specifies delivery address line case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case.
del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string
del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string
del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

### Return Code

negative for error, 0 for success

## addr\_get\_field

Use this function to retrieve validated address fields.

### Prototype

```
using ssa ;
```

```

public struct addr_get_field_struct addr_get_field (
int    suggest_idx , // Long in
int    field_idx    // Long in
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a field
field_idx	Specifies a field within the nth suggestion
field_value	The field value
field_val_status	Specifies how this field matched the validation data
field_val_mods	Specifies how this field was modified by validation data

## Return Code

negative for error, 0 for success

## addr\_get\_field\_count

Use this function to determine the max number of address fields.

### Prototype

```

using ssa ;
public int addr_get_field_count ( ) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of address fields

## Return Code

negative for error, 0 for success

## addr\_get\_field\_ext

Use this function to retrieve all getter fields.

### Prototype

```

using ssa ;

```

```

public byte [ ] addr_get_field_ext (
int    suggest_idx , // Long in
int    field_operation , // Long in
string field_name , // String in
int    field_item_line , // Long in
string field_type // String in
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get fields
field_operation	Field operation Option 0 for AddressElements Option 1 for AddressLines Option 2 for AddressComplete Option 3 for EnrichmentData Option 4 for ResultDataParameter Option 5 for EnrichmentDataStatus
field_name	Refer AD Result.dtd for field names
field_item_line	Represent field line number or field item number
field_type	Refer AD Result.dtd for field attribute Type
field_value	Cleansed field output

## Return Code

negative for error, 0 for success

## addr\_get\_field\_idx

Use this function to retrieve individual address fields.

### Prototype

```

using ssa ;
public byte [ ] addr_get_field_idx (
int    suggest_idx , // Long in
int    field_idx // Long in
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a field. On successful parse, use 0 for ASM for AD version 4, 1 for ASM for AD version 5

field_idx	Specifies a field within the nth suggestion
field_value	The field value

### Return Code

negative for error, 0 for success

## addr\_get\_field\_info\_ext

Use this function to retrieve information about a suggestion.

### Prototype

```
using ssa ;
public struct addr_get_field_info_ext_struct addr_get_field_info_ext (
int    suggest_idx    // Long in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each address field
addr_label_encoded	The returned label
addr_label_charset	The character set used in the address label
score	The returned label's score

### Return Code

negative for error, 0 for success

## addr\_get\_field\_len

Use this function to determine the max field length.

### Prototype

```
using ssa ;
public int addr_get_field_len ( ) ;
```



## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max address field length in bytes

## Return Code

negative for error, 0 for success

# addr\_get\_line\_len

## Prototype

```
using ssa ;  
public int addr_get_line_len ( ) ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max line length in bytes

## Return Code

negative for error, 0 for success

# addr\_get\_option

Use this function to set optional parameters.

## Prototype

```
using ssa ;  
public string addr_get_option (   
    string param // String in  
    ) ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

param	This field specifies the name of the option to get.
value	Returns the value for the option.

### Return Code

negative for error, 0 for success

## addr\_init

Use this function to initialize the Address Standardization interface.

### Prototype

```
using ssa ;
public void addr_init (
int max_memory / 7 Long in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_memory	This field specifies the maximum amount of memory (MB) to be used by the Address Standardization engine.

### Return Code

negative for error, 0 for success

## addr\_parse

Use this function to parse an address.

### Prototype

```
using ssa ;
public int [ ] addr_parse ( ) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_length	An array containing the length of each parsed field

### Return Code

negative for error, 0 for success

## addr\_preload\_country

Use this function to preload country database.

### Prototype

```
using ssa ;
public void addr_preload_country (
string preload_type , // String in
string preload_country , // String in
string val_mode // String in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
preload_type	type of preload to perform
preload_country	country database to be preloaded
val_mode	This field specified validation mode

### Return Code

negative for error, 0 for success

## addr\_set\_attrib

Use this function to specify the character set of the data and a default country.

### Prototype

```
using ssa ;
public void addr_set_attrib (
string char_set , // String in
string default_country // String in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
char_set	The name of the character set used to encode the input and output.
default_country	The default country used for validation when parsing cannot detect a country name.

### Return Code

negative for error, 0 for success

## addr\_set\_del\_lines

Use this function to set delivery address line information.

### Prototype

```
using ssa ;
public void addr_set_del_lines (
byte [ ] del_line1 , // Block in
byte [ ] del_line2 , // Block in
byte [ ] del_line3 , // Block in
byte [ ] del_line4 , // Block in
byte [ ] del_line5 , // Block in
byte [ ] del_line6 // Block in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
del_line1	delivery address line 1 input string
del_line2	delivery address line 2 input string
del_line3	delivery address line 3 input string
del_line4	delivery address line 4 input string
del_line5	delivery address line 5 input string
del_line6	delivery address line 6 input string

### Return Code

negative for error, 0 for success

## addr\_set\_field\_case

Use this function to set individual input fields case option.

### Prototype

```
using ssa ;
public void addr_set_field_case (
int field_idx , // Long in
int field_case // Long in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section

ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_case	Specifies output field case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case.

### Return Code

negative for error, 0 for success

## addr\_set\_field\_idx

Use this function to set individual input fields by idx.

### Prototype

```
using ssa ;
public void addr_set_field_idx (
    int      field_idx , // Long in
    byte [ ] field_value // Block in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_value	Specifies a value for the nth field

### Return Code

negative for error, 0 for success

## addr\_set\_field\_name

Use this function to set individual input fields by name.

### Prototype

```
using ssa ;
public void addr_set_field_name (
    string field_name , // String in
    byte [ ] field_value // Block in
) ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_name	Specifies the name of the field to set
field_value	Specifies a value for the field

## Return Code

negative for error, 0 for success

## addr\_set\_lines

Use this function to provide an address to parse or validate.

## Prototype

```
using ssa ;
public void addr_set_lines (
byte [ ] line_1 , // Block in
byte [ ] line_2 , // Block in
byte [ ] line_3 , // Block in
byte [ ] line_4 , // Block in
byte [ ] line_5 , // Block in
byte [ ] line_6 , // Block in
byte [ ] line_7 , // Block in
byte [ ] line_8 , // Block in
byte [ ] line_9 , // Block in
byte [ ] line_10 // Block in
) ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
line_1	The first line of the address
line_2	The second line of the address
line_3	The third line of the address
line_4	The fourth line of the address
line_5	The fifth line of the address
line_6	The sixth line of the address
line_7	The seventh line of the address

line_8	The eighth line of the address
line_9	The ninth line of the address
line_10	The tenth line of the address

### Return Code

negative for error, 0 for success

## addr\_set\_option

Use this function to set optional parameters.

### Prototype

```
using ssa ;
public void addr_set_option (
    string param, // String in
    string value // String in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to set.
value	This field specifies a value for the option.

### Return Code

negative for error, 0 for success

## addr\_validate

Use this function to validate an address.

### Prototype

```
using ssa ;
public struct addr_validate_struct addr_validate ( ) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

status	The status returned by the validation process
n_suggest	The number of suggestions generated by validation

### Return Code

negative for error, 0 for success

## close

Closes the SSA session and deallocates memory. Similarly to **ssa\_open**, it is optional.

### Prototype

```
using ssa ;
public void close (
    string controls // String in
) ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## convert\_keys

Used to convert 8 byte keys (from server) to 5 byte keys (for client).

### Prototype

```
using ssa ;
public byte [ ] [ ] convert_keys (
    string [ ] keys8 // StringArray in
) ;
```

### Parameters

keys8	block of 8 byte keys to be converted
keys5	block of 5 byte keys (output)



rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## convert\_ranges

Used to convert 8 byte ranges (from server) to 5 byte ranges (for client).

### Prototype

```
using ssa ;
public byte [ ] [ ] convert_ranges (
string [ ] ranges8 // StringArray in
) ;
```

### Parameters

ranges8	block of 8 byte ranges to be converted
ranges5	block of 5 byte ranges (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## disconnect

Releases resources allocated to a socket.

### Prototype

```
using ssa ;
public void disconnect ( ) ;
```

### Parameters

none

### Return Code

negative for error, 0 for success

## errors\_get\_all

Get the Server side error messages from the last API function that failed. This function should be called repeatedly until it returns 1, meaning all messages have been retrieved.

**Note:** If a communication (socket) error occurred, this function will also fail. Refer to the *OPERATIONS MANUAL, Error Log* section for information on interpreting the Error Log.

## Prototype

```
using ssa ;  
public string errors_get_all ( ) ;
```

## Parameters

msg is an error message.

## Return Code

negative for error, 0 for success

## get\_keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

## Prototype

```
using ssa ;  
public string [ ] get_keys (  
string controls , // String in  
string data // String in  
) ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

## Return Code

negative for error, 0 for success

## get\_keys\_encoded

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

## Prototype

```
using ssa ;
public string [ ] get_keys_encoded (
string controls , // String in
string data , // EncodedString in
int dataSize ,
string dataEncType
) ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

## Return Code

negative for error, 0 for success

## get\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

## Prototype

```
using ssa ;
public string [ ] get_ranges (
string controls , // String in
string data // String in
) ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section

rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

### Return Code

negative for error, 0 for success

## get\_ranges\_encoded

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```
using ssa ;
public string [ ] get_ranges_encoded (
    string controls , // String in
    string data , // EncodedString in
    int dataSize ,
    string dataEncType
) ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

### Return Code

negative for error, 0 for success

## info

Returns Information regarding the selected System and Population.

### Prototype

```
using ssa ;
public string [ ] info (
    string controls // String in
) ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	A number defining the number of rows in the Items Array
info	An array of 1024 byte rows, containing the data returned for the specific <b>ssan3_info</b> call

### Return Code

negative for error, 0 for success

## keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```
using ssa ;
public byte [ ] [ ] keys (
    string controls , // String in
    byte [ ] data , // Block in
    string encoding // String in
) ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section

rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field
count	A number defining the actual number of keys returned for this name or address
keys	An array of keys returned for this name or address

### Return Code

negative for error, 0 for success

## match

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

### Prototype

```
using ssa ;
public struct match_struct match (
    string controls , // String in
    string search , // String in
    string file // String in
) ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section.
file	Refer to the Common Parameters section.
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

## Return Code

negative for error, 0 for success

## match\_encoded

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

## Prototype

```
using ssa ;
public struct match_struct match_encoded (
    string controls , // String in
    string search , // EncodedString in
    int searchSize ,
    string searchEncType ,
    string file , // EncodedString in
    int file Size ,
    string fileEncType
) ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section.
file	Refer to the Common Parameters section.
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

## Return Code

negative for error, 0 for success

## open

An optional function, but its use will improve performance. It opens and initiates an SSA-NAME3 session in preparation.

## Prototype

```
using ssa ;
public void open (
string system , // String in
string population , // String in
string controls // String in
) ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

## Prototype

```
using ssa ;
public byte [ ] [ ] ranges (
string controls , // String in
byte [ ] data , // Block in
string encoding // String in
) ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section



encoding	The encoding used for the data field
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs returned for this name or address

### Return Code

negative for error, 0 for success

## Calling from C++

### Use of Destructors

C++, unlike JAVA and C# has destructors that are called whenever an instance of an object goes out of scope (or the delete operator is called on that object). This allows for any calls to resource releasing type functions (**ssan3\_close** and **ssan3\_disconnect** for example) to be made from an appropriate 'destructor'. Of course as soon as a class has an explicit destructor it must also have an explicit 'copy constructor' and 'assignment operator'. Once objects start getting copied around, it is easy to end up with two instance of an object with identical socket handles. When one of these objects goes out of scope the destructor is called and the socket connection is closed. When the second object tries to use the socket connection a conflict arises. Two ways to avoid this situation are as follows:

- To use some type of reference counting in the class so that the socket connection is only closed when the last object which used it goes out of scope.
- Secondly, make the copy constructor and assignment operator for the class 'private' (and without a body). Then any attempts to make copies of these objects will cause a compilation error which it may then be possible to code around.

### ssan3\_addr\_get\_cass\_field

Use this function to retrieve cass specific address fields.

#### Prototype

```

long ssan3_addr_get_cass_field (
long sockh ,
long * session , // Long io
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
long suggest_idx , // Long in
long field_idx , // Long in
char * field_value , // Block out
long field_value_size
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a cass field
field_idx	Specifies a cass field within the nth suggestion
field_value	The cass field value

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_cass\_field\_cnt

Use this function to determine the max number of cass address fields.

## Prototype

```
long ssan3_addr_get_cass_field_cnt (  
long sockh ,  
long * session , // Long i o  
char * rsp , // String out  
long rsp_size ,  
char * ssaMsg , // String out  
long ssaMsg_size ,  
long * count // Long out  
);
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of cass address fields

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_cass\_field\_info

Use this function to retrieve information about a suggestion.

## Prototype

```
long ssan3_addr_get_cass_field_info (  
long sockh ,
```

```

long * session , // Long i o
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
long suggest_idx , // Long in
long * field_length , // LongArray out
long field_length_num
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each cass address field

## Return Code

negative for error, 0 for success

Enter an example to illustrate your reference here (optional).

## ssan3\_addr\_get\_del\_lines

(deprecated) Use this function to retrieve delivery address line information This function is deprecated. Use **addr\_get\_del\_lines\_ext** instead.

## Prototype

```

long ssan3_addr_get_del_lines (
long sockh ,
long * session , // Long io
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
long suggest_idx , // Long in
char * del_line1 , // Block out
long del_line1_size ,
char * del_line2 , // Block out
long del_line2_size ,
char * del_line3 , // Block out
long del_line3_size ,
char * del_line4 , // Block out
long del_line4_size ,
char * del_line5 , // Block out
long del_line5_size ,
char * del_line6 , // Block out
long del_line6_size
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string
del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string
del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_del\_lines\_ext

Use this function to retrieve delivery address line information.

## Prototype

```
long ssan3_addr_get_del_lines_ext (
long sockh ,
long * session , // Long io
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
long suggest_idx , // Long in
long del_case , // Long in
char * del_line1 , // Block out
long del_line1_size ,
char * del_line2 , // Block out
long del_line2_size ,
char * del_line3 , // Block out
long del_line3_size ,
char * del_line4 , // Block out
long del_line4_size ,
char * del_line5 , // Block out
long del_line5_size ,
char * del_line6 , // Block out
long del_line6_size
) ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_case	Specifies delivery address line case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case.
del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string
del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string
del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field

Use this function to retrieve validated address fields.

## Prototype

```
long ssan3_addr_get_field (
long sockh ,
long * session ,          // Long io
char * rsp ,              // String out
long  rsp_size ,
char * ssaMsg ,          // String out
long  ssaMsg_size ,
long  suggest_idx ,      // Long in
long  field_idx ,        // Long in
char * field_value ,     // Block out
long  field_value_size ,
long * field_val_status , // Long out
long * field_val_mods    // Long out
) ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

suggest_idx	Specifies the nth suggestion from which to get a field
field_idx	Specifies a field within the nth suggestion
field_value	The field value
field_val_status	Specifies how this field matched the validation data
field_val_mods	Specifies how this field was modified by validation data

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_count

Use this function to determine the max number of address fields.

### Prototype

```

long ssan3_addr_get_field_count (
    long sockh ,
    long * session , // Long io
    char * rsp , // String out
    long  rsp_size ,
    char * ssaMsg , // String out
    long  ssaMsg_size ,
    long * count // Long out
) ;

```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of address fields

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_ext

Use this function to retrieve all getter fields.

### Prototype

```

long ssan3_addr_get_field_ext (
    long sockh ,
    long * session , // Long io
    char * rsp , // String out
    long  rsp_size ,
    char * ssaMsg , // String out
    long  ssaMsg_size ,
    long  suggest_idx , // Long in

```

```

long field_operation , // Long in
char * field_name , // String in
long field_item_line , // Long in
char * field_type , // String in
char * field_value , // Block out
long field_value_size
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get fields
field_operation	Field operation Option 0 for AddressElements Option 1 for AddressLines Option 2 for AddressComplete Option 3 for EnrichmentData Option 4 for ResultDataParameter Option 5 for EnrichmentDataStatus
field_name	Refer AD Result.dtd for field names
field_item_line	Represent field line number or field item number
field_type	Refer AD Result.dtd for field attribute Type
field_value	Cleansed field output

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_idx

Use this function to retrieve individual address fields.

## Prototype

```

long ssan3_addr_get_field_idx (
long sockh ,
long * session , // Long io
char * rsp , // String outlong rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
long suggest_idx , // Long in
long field_idx , // Long in
char * field_value , // Block out
long field_value_size
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

suggest_idx	Specifies the nth suggestion from which to get a field. On successful parse, use 0 for ASM for AD version 4, 1 for ASM for AD version 5
field_idx	Specifies a field within the nth suggestion
field_value	The field value

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_info\_ext

Use this function to retrieve information about a suggestion.

### Prototype

```

long ssan3_addr_get_field_info_ext (
long sockh ,
long * session , // Long io
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
long suggest_idx , // Long in
long * field_length , // LongArray out
long field_length_num ,
char * addr_label_encoded , // Block out
long addr_label_encoded_size ,
char * addr_label_charset , // String out
long addr_label_charset_size ,
long * score // Long out
) ;

```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each address field
addr_label_encoded	The returned label
addr_label_charset	The character set used in the address label
score	The returned label's score

### Return Code

negative for error, 0 for success



## ssan3\_addr\_get\_field\_len

Use this function to determine the max field length.

### Prototype

```
long ssan3_addr_get_field_len (  
    long sockh ,  
    long * session , // Long io  
    char * rsp , // String out  
    long rsp_size ,  
    char * ssaMsg , // String out  
    long ssaMsg_size ,  
    long * max_len // Long out  
);
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max address field length in bytes

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_line\_len

### Prototype

```
long ssan3_addr_get_line_len (  
    long sockh ,  
    long * session , // Long io  
    char * rsp , // String out  
    long rsp_size ,  
    char * ssaMsg , // String out  
    long ssaMsg_size ,  
    long * max_len // Long out  
);
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max line length in bytes

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_option

Use this function to set optional parameters

### Prototype

```
long ssan3_addr_get_option (
long sockh ,
long * session , // Long io
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
char * param, // String in
char * value , // String out
long value_size
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to get.
value	Returns the value for the option.

### Return Code

negative for error, 0 for success

## ssan3\_addr\_init

Use this function to initialize the Address Standardization interface.

### Prototype

```
long ssan3_addr_init (
long sockh ,
long * session , // Long io
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
long max_memory // Long in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section

ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_memory	This field specifies the maximum amount of memory (MB) to be used by the Address Standarization engine.

### Return Code

negative for error, 0 for success

## ssan3\_addr\_parse

Use this function to parse an address.

### Prototype

```

long ssan3_addr_parse (
long sockh ,
long * session , // Long io
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
long * field_length , // LongArray out
long field_length_num
) ;

```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_length	An array containing the length of each parsed field

### Return Code

negative for error, 0 for success

## ssan3\_addr\_preload\_country

Use this function to preload country database.

### Prototype

```

long ssan3_addr_preload_country (
long sockh ,
long * session , // Long io
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
char * preload_type , // String in
char * preload_country , // String in
char * val_mode // String in
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
preload_type	type of preload to perform
preload_country	country database to be preloaded
val_mode	This field specified validation mode

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_attrb

Use this function to specify the character set of the data and a default country.

## Prototype

```
long ssan3_addr_set_attrb (  
    long sockh ,  
    long * session , // Long io  
    char * rsp , // String out  
    long rsp_size ,  
    char * ssaMsg , // String out  
    long ssaMsg_size ,  
    char * char_set , // String in  
    char * default_country // String in  
);
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
char_set	The name of the character set used to encode the input and output
default_country	The default country used for validation when parsing cannot detect a country name

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_del\_lines

Use this function to set delivery address line information.

## Prototype

```
long ssan3_addr_set_del_lines (  
long sockh ,  
long * session , // Long io  
char * rsp , // String out  
long rsp_size ,  
char * ssaMsg , // String out  
long ssaMsg_size ,  
char * del_line1 , // Block in  
long del_line1_size ,  
char * del_line2 , // Block in  
long del_line2_size ,  
char * del_line3 , // Block in  
long del_line3_size ,  
char * del_line4 , // Block in  
long del_line4_size ,  
char * del_line5 , // Block in  
long del_line5_size ,  
char * del_line6 , // Block in  
long del_line6_size  
);
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
del_line1	delivery address line 1 input string
del_line2	delivery address line 2 input string
del_line3	delivery address line 3 input string
del_line4	delivery address line 4 input string
del_line5	delivery address line 5 input string
del_line6	delivery address line 6 input string

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_field\_case

Use this function to set individual input fields case option.

## Prototype

```
long ssan3_addr_set_field_case (  
long sockh ,  
long * session , // Long io  
char * rsp , // String out  
long rsp_size ,  
char * ssaMsg , // String out  
long ssaMsg_size ,  
long field_idx , // Long in
```

```

    long field_case // Long in
);

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_case	Specifies output field case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case.

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_field\_idx

Use this function to set individual input fields by idx.

### Prototype

```

long ssan3_addr_set_field_idx (
long sockh ,
long * session , // Long io
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
long field_idx , // Long in
char * field_value , // Block in
long field_value_size
);

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_value	Specifies a value for the nth field

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_field\_name

Use this function to set individual input fields by name.

## Prototype

```
long ssan3_addr_set_field_name (  
long sockh ,  
long * session , // Long io  
char * rsp , // String out  
long rsp_size ,  
char * ssaMsg , // String out  
long ssaMsg_size ,  
char * field_name , // String in  
char * field_value , // Block in  
long field_value_size  
);
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_name	Specifies the name of the field to set
field_value	Specifies a value for the field

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_lines

Use this function to provide an address to parse or validate.

## Prototype

```
long ssan3_addr_set_lines (  
long sockh ,  
long * session , // Long io  
char * rsp , // String out  
long rsp_size ,  
char * ssaMsg , // String out  
long ssaMsg_size ,  
char * line_1 , // Block in  
long line_1_size ,  
char * line_2 , // Block in  
long line_2_size ,  
char * line_3 , // Block in  
long line_3_size ,  
char * line_4 , // Block in  
long line_4_size ,  
char * line_5 , // Block in  
long line_5_size ,  
char * line_6 , // Block in  
long line_6_size ,  
char * line_7 , // Block in  
long line_7_size ,  
char * line_8 , // Block in  
long line_8_size ,  
char * line_9 , // Block in  
long line_9_size ,  
char * line_10 , // Block in
```

```

    long   line_10_size
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
line_1	The first line of the address
line_2	The second line of the address
line_3	The third line of the address
line_4	The fourth line of the address
line_5	The fifth line of the address
line_6	The sixth line of the address
line_7	The seventh line of the address
line_8	The eighth line of the address
line_9	The ninth line of the address
line_10	The tenth line of the address

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_option

Use this function to set optional parameters.

### Prototype

```

long ssan3_addr_set_option (
long sockh ,
long * session , // Long io
char * rsp , // String out
long rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
char * param, // String in
char * value // String in
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section



ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to set.
value	This field specifies a value for the option.

### Return Code

negative for error, 0 for success

## ssan3\_addr\_validate

Use this function to validate an address.

### Prototype

```

long ssan3_addr_validate (
long sockh ,
long * session ,          // Long io
char * rsp ,              // String out
long  rsp_size ,
char * ssaMsg ,          // String out
long  ssaMsg_size ,
long * status ,          // Long out
long * n_suggest // Long out
) ;

```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
status	The status returned by the validation process
n_suggest	The number of suggestions generated by validation

### Return Code

negative for error, 0 for success

## ssan3\_close

Closes the SSA session and deallocates memory. Similarly to **ssa\_open**, it is optional.

### Prototype

```

long ssan3_close (
long sockh ,
long * session ,          // Long io
char * system ,          // String in
char * population ,      // String in
char * controls ,        // String in
char * rsp ,              // String out
long  rsp_size ,
char * ssaMsg ,          // String out

```

```

    long ssaMsg_size
);

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## ssan3\_connect

Initiates a socket.

### Prototype

```

long ssan3_connect (
char * host , // String in
long port , // Long in
long * sockh // Long out
);

```

## Parameters

host	This is the host to connect to.
port	This is the port to connect to.
sockh	This is a socket handle.

## Return Code

negative for error, 0 for success

## ssan3\_convert\_keys

Used to convert 8 byte keys (from server) to 5 byte keys (for client).

### Prototype

```

long ssan3_convert_keys (
long sockh ,
char ** keys8 , // StringArray in
long keys8_num ,
char ** keys5 , // BlockArray out
long keys5_num ,
long keys5_size ,

```

```

char *  rsp ,          // String out
long   rsp_size ,
char *  ssaMsg ,      // String out
long   ssaMsg_size
) ;

```

### Parameters

keys8	Block of 8 byte keys to be converted
keys5	Block of 5 byte keys (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## ssan3\_convert\_ranges

Used to convert 8 byte ranges (from server) to 5 byte ranges (for client).

### Prototype

```

long ssan3_convert_ranges (
long sockh ,
char ** ranges8 , // StringArray in
long  ranges8_num ,
char ** ranges5 , // BlockArray out
long  ranges5_num ,
long  ranges5_size ,
char *  rsp , // String out
long  rsp_size ,
char *  ssaMsg , // String out
long  ssaMsg_size
) ;

```

### Parameters

ranges8	Block of 8 byte ranges to be converted
ranges5	Block of 5 byte ranges (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## ssan3\_disconnect

Releases resources allocated to a socket.

## Prototype

```
long ssan3_disconnect (
    long sockh
) ;
```

## Parameters

none

## Return Code

negative for error, 0 for success

## ssan3\_errors\_get\_all

Get the Server side error messages from the last API function that failed. This function should be called repeatedly until it returns 1, meaning all messages have been retrieved.

**Note:** if a communication (socket) error occurred, this function will also fail. Refer to the *OPERATIONS MANUAL, Error Log* section for information on interpreting the Error Log.

## Prototype

```
long ssan3_errors_get_all (
    long sockh ,
    char * msg , // String out
    long msg_size
) ;
```

## Parameters

msg is an error message.

## Return Code

negative for error, 0 for success

## ssan3\_get\_keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

## Prototype

```
long ssan3_get_keys (
    long sockh ,
    long * session , // Long io
    char * system , // String in
    char * population , // String in
    char * cont rol s , // String in
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    char * data , // String in
    long * count , // Long out
    char ** keys , // StringArray out
    long keys_size
) ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

## Return Code

negative for error, 0 for success

## ssan3\_get\_keys\_encoded

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

## Prototype

```
long ssan3_get_keys_encoded (
long sockh ,
long * session ,          // Long i o
char * system ,          // String in
char * population ,      // String in
char * cont rol s ,      // String in
char * rsp ,              // String out
long  rsp_size ,
char * ssaMsg ,          // String out
long  ssaMsg_size ,
char * data ,            // EncodedString in
long  dataSize ,
char * dataEncType ,
long * count ,           // Long out
char ** keys ,           // StringArray out
long  keys_size
) ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section

controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

### Return Code

negative for error, 0 for success

Enter an example to illustrate your reference here (optional).

## ssan3\_get\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```

long ssan3_get_ranges (
long sockh ,
long * session ,           // Long io
char * system ,           // String in
char * population ,       // String in
char * controls ,         // String in
char * rsp ,              // String out
long  rsp_size ,
char * ssaMsg ,           // String out
long  ssaMsg_size ,
char * data ,             // String in
long * count ,            // Long out
char ** stab ,           // StringArray out
long  stab_size
) ;

```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section

count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

### Return Code

negative for error, 0 for success

## ssan3\_get\_ranges\_encoded

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```

long ssan3_get_ranges_encoded (
long sockh ,
long * session , // Long io
char * system , // String in
char * population , // String in
char * controls , // String in
char * rsp , // String out
long  rsp_size ,
char * ssaMsg , // String out
long ssaMsg_size ,
char * data , // EncodedString in
long dataSize ,
char * dataEncType ,
long * count , // Long out
char ** stab , // StringArray out
long stab_size
) ;

```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

### Return Code

negative for error, 0 for success

## ssan3\_info

Returns Information regarding the selected System and Population.

### Prototype

```
long ssan3_info (
    long sockh ,
    long * session , // Long io
    char * system , // String in
    char * population , // String in
    char * controls , // String in
    char * rsp , // String out
    long  rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    long * count , // Long out
    char ** info , // StringArray out
    long info_size
) ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	A number defining the number of rows in the Items Array
info	An array of 1024 byte rows, containing the data returned for the specific <b>ssan3_info</b> call

### Return Code

negative for error, 0 for success

## ssan3\_keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```
long ssan3_keys (
    long sockh ,
    long * session , // Long io
    char * system , // String in
    char * population , // String in
    char * controls , // String in
    char * rsp , // String out
    long  rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    char * data , // Block in
)
```



```

        long data_size ,
        char * encoding , // String in
        long * count , // Long out
        char ** keys , // BlockArray out
        long keys_num,
        long keys_size
    ) ;

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field
count	A number defining the actual number of keys returned for this name or address
keys	An array of keys returned for this name or address

## Return Code

negative for error, 0 for success

## ssan3\_match

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

## Prototype

```

long ssan3_match (
    long sockh ,
    long * session , // Long io
    char * system , // String in
    char * population , // String in
    char * controls , // String in
    char * rsp , // String out
    long  rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    char * search , // String in
    char * file , // String in
    char * score , // String out
    long  score_size ,
    char * decision , // String out
    long decision_size
) ;

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to Common Parameters section
file	Refer to Common Parameters section
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

## Return Code

negative for error, 0 for success

## ssan3\_match\_encoded

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

## Prototype

```
long ssan3_match_encoded (
    long sockh ,
    long * session , // Long io
    char * system , // String in
    char * population , // String in
    char * controls , // String in
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    char * search , // EncodedString in
    long searchSize ,
    char * searchEncType ,
    char * file , // EncodedString in
    long fileSize ,
    char * fileEncType ,
    char * score , // String out
    long score_size ,
    char * decision , // String out
    long decision_size
) ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section.
file	Refer to the Common Parameters section.
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

## Return Code

negative for error, 0 for success

## ssan3\_open

An optional function, but its use will improve performance. It opens and initiates an SSA-NAME3 session in preparation.

## Prototype

```
long ssan3_open (
    long sockh ,
    long * session , // Long io
    char * system , // String in
    char * population , // String in
    char * controls , // String in
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size
) ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section

rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## ssan3\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a `Select` statement to retrieve records from the SSA Keys table.

### Prototype

```

long ssan3_ranges (
    long sockh ,
    long * session , // Long i o
    char * system , // String in
    char * population , // String in
    char * controls , // String in
    char * rsp , // String out
    long rsp_size ,
    char * ssaMsg , // String out
    long ssaMsg_size ,
    char * data , // Block in
    long data_size ,
    char * encoding , // String in
    long * count , // Long out
    char ** stab , // BlockArray out
    long stab_num ,
    long stab_size
) ;

```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs returned for this name or address

### Return Code

negative for error, 0 for success

# Calling from COBOL

## SSAN3-ADDR-GET-CASS-FIELD

Use this function to retrieve cass specific address fields.

### Prototype

```
CALL SSAN3-ADDR-GET-CASS-FIELD USING
    BY VALUE SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY VALUE SUGGEST-IDX,
    BY VALUE FIELD-IDX,
    BY REFERENCE FIELD-VALUE,
    BY VALUE FIELD-VALUE-SIZE
END CALL
```

### Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
SUGGEST-IDX	Specifies the nth suggestion from which to get a cass field
FIELD-IDX	Specifies a cass field within the nth suggestion
FIELD-VALUE	The cass field value

### Return Code

negative for error, 0 for success

## SSAN3-ADDR-GET-CASS-FIELD-CNT

Use this function to determine the max number of cass address fields.

### Prototype

```
CALL SSAN3-ADDR-GET-CASS-FIELD-CNT USING
    BY VALUE SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY REFERENCE COUNT
END CALL
```

### Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section

SSAMSG	Refer to the SSA-NAME3 Error Messages section
COUNT	Returns the max number of cass address fields

### Return Code

negative for error, 0 for success

## SSAN3-ADDR-GET-CASS-FIELD-INFO

Use this function to retrieve information about a suggestion.

### Prototype

```
CALL SSAN3-ADDR-GET-CASS-FIELD-INFO USING
    BY VALUE SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY VALUE SUGGEST-IDX,
    BY REFERENCE FIELD-LENGTH,
    BY VALUE FIELD-LENGTH-NUM
END CALL
```

### Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
SUGGEST-IDX	Specifies the suggestion from which to retrieve information
FIELD-LENGTH	An array containing the length of each cass address field

### Return Code

negative for error, 0 for success

## SSAN3-ADDR-GET-DEL-LINES

(deprecated) Use this function to retrieve delivery address line information. This function is deprecated. Use **addr\_get\_del\_lines\_ext** instead

### Prototype

```
CALL SSAN3-ADDR-GET-DEL-LINES USING
    BY VALUE     SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY VALUE     SUGGEST-IDX,
    BY REFERENCE DEL-LINE1,
    BY VALUE     DEL-LINE1-SIZE,
    BY REFERENCE DEL-LINE2,
    BY VALUE     DEL-LINE2-SIZE,
    BY REFERENCE DEL-LINE3,
```

```

        BY VALUE      DEL-LINE3-SIZE,
        BY REFERENCE DEL-LINE4,
        BY VALUE      DEL-LINE4-SIZE,
        BY REFERENCE DEL-LINE5,
        BY VALUE      DEL-LINE5-SIZE,
        BY REFERENCE DEL-LINE6,
        BY VALUE      DEL-LINE6-SIZE
    END CALL

```

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
SUGGEST- IDX	Specifies the suggestion from which to get delivery address lines
DEL-LINE1	delivery address line 1 output string
DEL-LINE2	delivery address line 2 output string
DEL-LINE3	delivery address line 3 output string
DEL-LINE4	delivery address line 4 output string
DEL-LINE5	delivery address line 5 output string
DEL-LINE6	delivery address line 6 output string

## Return Code

negative for error, 0 for success

# SSAN3-ADDR-GET-DEL-LINES-EXT

Use this function to retrieve delivery address line information.

## Prototype

```

CALL SSAN3-ADDR-GET-DEL-LINES-EXT USING
    BY VALUE SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY VALUE SUGGEST-IDX,
    BY VALUE DEL-CASE,
    BY REFERENCE DEL-LINE1,
    BY VALUE DEL-LINE1-SIZE,
    BY REFERENCE DEL-LINE2,
    BY VALUE DEL-LINE2-SIZE,
    BY REFERENCE DEL-LINE3,
    BY VALUE DEL-LINE3-SIZE,
    BY REFERENCE DEL-LINE4,
    BY VALUE DEL-LINE4-SIZE,
    BY REFERENCE DEL-LINE5,
    BY VALUE DEL-LINE5-SIZE,
    BY REFERENCE DEL-LINE6,
    BY VALUE DEL-LINE6-SIZE
END CALL

```

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
SUGGEST-IDX	Specifies the suggestion from which to get delivery address lines
DEL-CASE	Specifies delivery address line case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case.
DEL-LINE1	delivery address line 1 output string
DEL-LINE2	delivery address line 2 output string
DEL-LINE3	delivery address line 3 output string
DEL-LINE4	delivery address line 4 output string
DEL-LINE5	delivery address line 5 output string
DEL-LINE6	delivery address line 6 output string

## Return Code

negative for error, 0 for success

## SSAN3-ADDR-GET-FIELD

Use this function to retrieve validated address fields.

## Prototype

```
CALL SSAN3-ADDR-GET-FIELD USING
    BY VALUE SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY VALUE SUGGEST-IDX,
    BY VALUE FIELD-IDX,
    BY REFERENCE FIELDVALUE,
    BY VALUE FIELD-VALUE-SIZE,
    BY REFERENCE FIELD-VAL-STATUS,
    BY REFERENCE FIELD-VAL-MODS

END CALL
```

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
SUGGEST-IDX	Specifies the nth suggestion from which to get a field



FIELD-IDX	Specifies a field within the nth suggestion
FIELD-VALUE	The field value
FIELD-VAL-STATUS	Specifies how this field matched the validation data
FIELD-VAL-MODS	Specifies how this field was modified by validation data

### Return Code

negative for error, 0 for success

## SSAN3-ADDR-GET-FIELD-COUNT

Use this function to determine the max number of address fields.

### Prototype

```
CALL SSAN3-ADDR-GET-FIELD-COUNT USING
    BY VALUE SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY REFERENCE COUNT
END CALL
```

### Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
COUNT	Returns the max number of address fields

### Return Code

negative for error, 0 for success

## SSAN3-ADDR-GET-FIELD-EXT

Use this function to retrieve all getter fields.

### Prototype

```
CALL SSAN3-ADDR-GET-FIELD-EXT USING
    BY VALUE SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY VALUE SUGGEST-IDX,
    BY VALUE FIELD-OPERATION,
    BY REFERENCE FIELD-NAME,
    BY VALUE FIELD-ITEM-LINE,
    BY REFERENCE FIELD-TYPE,
```

```

BY REFERENCE FIELD-VALUE,
BY VALUE FIELD-VALUE-SIZE
END CALL

```

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
SUGGEST-IDX	Specifies the suggestion from which to get fields
FIELD-OPERATION	Field operation Option 0 for AddressElements Option 1 for AddressLines Option 2 for AddressComplete Option 3 for EnrichmentData Option 4 for ResultDataParameter Option 5 for EnrichmentDataStatus
FIELD-NAME	Refer AD Result.dtd for field names
FIELD-ITEM-LINE	Represent field line number or field item number
FIELD-TYPE	Refer AD Result.dtd for field attribute Type
FIELD-VALUE	Cleansed field output

## Return Code

negative for error, 0 for success

## SSAN3-ADDR-GET-FIELD-IDX

Use this function to retrieve individual address fields.

## Prototype

```

CALL SSAN3-ADDR-GET-FIELD-IDX USING
BY VALUE SOCKH,
BY REFERENCE SESSION,
BY REFERENCE RSP ,
BY REFERENCE SSAMSG,
BY VALUE SUGGEST-IDX,
BY VALUE FIELD-IDX,
BY REFERENCE FIELD-VALUE,
BY VALUE FIELD-VALUE-SIZE
END CALL

```

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
SUGGEST-IDX	Specifies the nth suggestion from which to get a field. On successful parse, use 0 for ASM for AD version 4, 1 for ASM for AD version 5

FIELD-IDX	Specifies a field within the nth suggestion
FIELD-VALUE	The field value

### Return Code

negative for error, 0 for success

## SSAN3-ADDR-GET-FIELD-INFO-EXT

Use this function to retrieve information about a suggestion.

### Prototype

```
CALL SSAN3-ADDR-GET-FIELD-INFO-EXT USING
    BY VALUE SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY VALUE SUGGEST-IDX,
    BY REFERENCE FIELD-LENGTH,
    BY VALUE FIELD-LENGTH-NUM,
    BY REFERENCE ADDR-LABEL-ENCODED,
    BY VALUE ADDR-LABEL-ENCODED-SIZE,
    BY REFERENCE ADDR-LABEL-CHARSET,
    BY REFERENCE SCORE

END CALL
```

### Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
SUGGEST-IDX	Specifies the suggestion from which to retrieve information
FIELD-LENGTH	An array containing the length of each address field
ADDR-LABEL-ENCODED	The returned label
ADDR-LABEL-CHARSET	The character set used in the address label
SCORE	The returned label's score

### Return Code

negative for error, 0 for success

## SSAN3-ADDR-GET-FIELD-LEN

Use this function to determine the max field length.

## Prototype

```
CALL SSAN3-ADDR-GET-FIELD-LEN USING
                                     BY VALUE SOCKH,
                                     BY REFERENCE SESSION,
                                     BY REFERENCE RSP ,
                                     BY REFERENCE SSAMSG,
                                     BY REFERENCE MAX-LEN
END CALL
```

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
MAX-LEN	Returns the max address field length in bytes

## Return Code

negative for error, 0 for success

# SSAN3-ADDR-GET-LINE-LEN

## Prototype

```
CALL SSAN3-ADDR-GET-LINE-LEN USING
                                     BY VALUE SOCKH,
                                     BY REFERENCE SESSION,
                                     BY REFERENCE RSP ,
                                     BY REFERENCE SSAMSG,
                                     BY REFERENCE MAX-LEN
END CALL
```

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
MAX-LEN	Returns the max line length in bytes

## Return Code

negative for error, 0 for success

# SSAN3-ADDR-GET-OPTION

Use this function to set optional parameters.

## Prototype

```
CALL SSAN3-ADDR-GET-OPTION USING
```

```

BY VALUE      SOCKH,
BY REFERENCE  SESSION,
BY REFERENCE  RSP ,
BY REFERENCE  SSAMSG,
BY REFERENCE  PARAM,
BY REFERENCE  VALUE

END CALL

```

### Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
PARAM	This field specifies the name of the option to get.
VALUE	Returns the value for the option

### Return Code

negative for error, 0 for success

## SSAN3-ADDR-INIT

Use this function to initialize the Address Standardization interface.

### Prototype

```

CALL SSAN3-ADDR-INIT USING
BY VALUE SOCKH,
BY REFERENCE SESSION,
BY REFERENCE RSP ,
BY REFERENCE SSAMSG,
BY VALUE MAX-MEMORY

END CALL

```

### Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
MAX-MEMORY	This field specifies the maximum amount of memory (MB) to be used by the Address Standardization engine.

### Return Code

negative for error, 0 for success

## SSAN3-ADDR-PARSE

Use this function to parse an address.

## Prototype

```
CALL SSAN3-ADDR-PARSE USING
                                BY VALUE SOCKH,
                                BY REFERENCE SESSION,
                                BY REFERENCE RSP ,
                                BY REFERENCE SSAMSG,
                                BY REFERENCE FIELD-LENGTH,
                                BY VALUE FIELD-LENGTH-NUM
END CALL
```

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
FIELD-LENGTH	An array containing the length of each parsed field

## Return Code

negative for error, 0 for success

# SSAN3-ADDR-PRELOAD-COUNTRY

Use this function to preload country database.

## Prototype

```
CALL SSAN3-ADDR-PRELOAD-COUNTRY USING
                                BY VALUE SOCKH,
                                BY REFERENCE SESSION,
                                BY REFERENCE RSP ,
                                BY REFERENCE SSAMSG,
                                BY REFERENCE PRELOAD-TYPE,
                                BY REFERENCE PRELOAD-COUNTRY,
                                BY REFERENCE VAL-MODE
END CALL
```

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
PRELOAD-TYPE	Type of preload to perform
PRELOAD-COUNTRY	Country database to be preloaded
VAL-MODE	This field specified validation mode

## Return Code

negative for error, 0 for success

## SSAN3-ADDR-SET-ATTRIB

Use this function to specify the character set of the data and a default country.

### Prototype

```
CALL SSAN3-ADDR-SET-ATTRIB USING
                                BY VALUE      SOCKH,
                                BY REFERENCE  SESSION,
                                BY REFERENCE  RSP ,
                                BY REFERENCE  SSAMSG,
                                BY REFERENCE  CHAR-SET,
                                BY REFERENCE  DEFAULT-COUNTRY
END CALL
```

### Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
CHAR-SET	The name of the character set used to encode the input and output.
DEFAULT-COUNTRY	The default country used for validation when parsing cannot detect a country name.

### Return Code

negative for error, 0 for success

## SSAN3-ADDR-SET-DEL-LINES

Use this function to set delivery address line information.

### Prototype

```
CALL SSAN3-ADDR-SET-DEL-LINES USING
                                BY VALUE      SOCKH,
                                BY REFERENCE  SESSION,
                                BY REFERENCE  RSP ,
                                BY REFERENCE  SSAMSG,
                                BY REFERENCE  DEL-LINE1,
                                BY REFERENCE  DEL-LINE2,
                                BY REFERENCE  DEL-LINE3,
                                BY REFERENCE  DEL-LINE4,
                                BY REFERENCE  DEL-LINE5,
                                BY REFERENCE  DEL-LINE6
END CALL
```

### Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
DEL-LINE1	Delivery address line 1 input string

DEL-LINE2	Delivery address line 2 input string
DEL-LINE3	Delivery address line 3 input string
DEL-LINE4	Delivery address line 4 input string
DEL-LINE5	Delivery address line 5 input string

### Return Code

negative for error, 0 for success

## SSAN3-ADDR-SET-FIELD-CASE

Use this function to set individual input fields case option.

### Prototype

```
CALL SSAN3-ADDR-SET-FIELD-CASE USING
    BY VALUE SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY VALUE FIELD-IDX,
    BY VALUE FIELD-CASE
END CALL
```

### Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
FIELD-IDX	Specifies the nth field to set
FIELD-CASE	Specifies output field case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case.

### Return Code

negative for error, 0 for success

## SSAN3-ADDR-SET-FIELD-IDX

Use this function to set individual input fields by idx.

### Prototype

```
CALL SSAN3-ADDR-SET-FIELD-IDX USING
    BY VALUE SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY VALUE FIELD-IDX,
```



BY REFERENCE FIELD-VALUE

END CALL

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
FIELD-IDX	Specifies the nth field to set
FIELD-VALUE	Specifies a value for the nth field

## Return Code

negative for error, 0 for success

# SSAN3-ADDR-SET-FIELD-NAME

Use this function to set individual input fields by name.

## Prototype

```
CALL SSAN3-ADDR-SET-FIELD-NAME USING
    BY VALUE SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY REFERENCE FIELD-NAME,
    BY REFERENCE FIELD-VALUE
END CALL
```

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
FIELD-NAME	Specifies the name of the field to set
FIELD-VALUE	Specifies a value for the field

## Return Code

negative for error, 0 for success

# SSAN3-ADDR-SET-LINES

Use this function to provide an address to parse or validate.

## Prototype

```
CALL SSAN3-ADDR-SET-LINES USING
```

```

BY VALUE     SOCKH,
BY REFERENCE SESSION,
BY REFERENCE RSP ,
BY REFERENCE SSAMSG,
BY REFERENCE LINE-1 ,
BY REFERENCE LINE-2 ,
BY REFERENCE LINE-3 ,
BY REFERENCE LINE-4 ,
BY REFERENCE LINE-5 ,
BY REFERENCE LINE-6 ,
BY REFERENCE LINE-7 ,
BY REFERENCE LINE-8 ,
BY REFERENCE LINE-9 ,
BY REFERENCE LINE-10

```

END CALL

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
LINE-1	The first line of the address
LINE-2	The second line of the address
LINE-3	The third line of the address
LINE-4	The fourth line of the address
LINE-5	The fifth line of the address
LINE-6	The sixth line of the address
LINE-7	The seventh line of the address
LINE-8	The eighth line of the address
LINE-9	The ninth line of the address
LINE-10	The tenth line of the address

## Return Code

negative for error, 0 for success

## SSAN3-ADDR-SET-OPTION

Use this function to set optional parameters.

## Prototype

```

CALL SSAN3-ADDR-SET-OPTION USING
BY VALUE SOCKH,
BY REFERENCE SESSION,
BY REFERENCE RSP ,
BY REFERENCE SSAMSG,
BY REFERENCE PARAM,

```

BY REFERENCE VALUE

END CALL

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
PARAM	This field specifies the name of the option to set.
VALUE	This field specifies a value for the option.

## Return Code

negative for error, 0 for success

# SSAN3-ADDR-VALIDATE

Use this function to validate an address.

## Prototype

```
CALL SSAN3-ADDR-VALIDATE USING
                                BY VALUE           SOCKH,
                                BY REFERENCE SESSION,
                                BY REFERENCE RSP ,
                                BY REFERENCE SSAMSG,
                                BY REFERENCE STATUS,
                                BY REFERENCE N-SUGGEST
END CALL
```

## Parameters

SESSION	Refer to the Common Parameters section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
STATUS	The status returned by the validation process
N-SUGGEST	The number of suggestions generated by validation

## Return Code

negative for error, 0 for success

# SSAN3-CLOSE

Closes the SSA session and deallocates memory. Similarly to **ssa\_open**, it is optional.

## Prototype

```
CALL SSAN3-CLOSE USING
```

```

BY VALUE SOCKH,
BY REFERENCE SESSION,
BY REFERENCE SYSTEM,
BY REFERENCE POPULATION,
BY REFERENCE CONTROLS,
BY REFERENCE RSP ,
BY REFERENCE SSAMSG

END CALL

```

### Parameters

SESSION	Refer to the Common Parameters section
SYSTEM	Refer to the Common Parameters section
POPULATION	Refer to the Common Parameters section
CONTROLS	Refer to the Controls section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## SSAN3-CONNECT

Initiates a socket.

### Prototype

```

CALL SSAN3-CONNECT USING
BY REFERENCE HOST,
BY VALUE PORT,
BY REFERENCE SOCKH

END CALL

```

### Parameters

HOST	This is the host to connect to.
PORT	This is the port to connect to.
SOCKH	This is a socket handle.

### Return Code

negative for error, 0 for success

## SSAN3-CONVERT-KEYS

Used to convert 8 byte keys (from server) to 5 byte keys (for client).

### Prototype

```

CALL SSAN3-CONVERT-KEYS USING

```

```

BY VALUE SOCKH,
BY REFERENCE KEYS8 ,
BY REFERENCE KEYS5 ,
BY VALUE KEYS5-NUM,
BY VALUE KEYS5-SIZE ,
BY REFERENCE RSP ,
BY REFERENCE SSAMSG

END CALL

```

### Parameters

KEYS8	Block of 8 byte keys to be converted
KEYS5	Block of 5 byte keys (output)
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## SSAN3-CONVERT-RANGES

Used to convert 8 byte ranges (from server) to 5 byte ranges (for client).

### Prototype

```

CALL SSAN3-CONVERT-RANGES USING
BY VALUE      SOCKH,
BY REFERENCE  RANGES8,
BY REFERENCE  RANGES5,
BY VALUE      RANGES5-NUM,
BY VALUE      RANGES5-SIZE,
BY REFERENCE  RSP ,
BY REFERENCE  SSAMSG

END CALL

```

### Parameters

RANGES8	Block of 8 byte ranges to be converted
RANGES5	Block of 5 byte ranges (output)
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## SSAN3-DISCONNECT

Releases resources allocated to a socket.

## Prototype

```
CALL SSAN3-DISCONNECT USING
    BY VALUE SOCKH
END CALL
```

## Parameters

none

## Return Code

negative for error, 0 for success

# SSAN3-ERRORS-GET-ALL

Get the Server side error messages from the last API function that failed. This function should be called repeatedly until it returns 1, meaning all messages have been retrieved.

**Note:** If a communication (socket) error occurred, this function will also fail. Refer to the *OPERATIONS MANUAL*, *Error Log* section for information on interpreting the Error Log.

## Prototype

```
CALL SSAN3-ERRORS-GET-ALL USING
    BY VALUE SOCKH,
    BY REFERENCE MSG
END CALL
```

## Parameters

MSG	This is an error message.
-----	---------------------------

## Return Code

negative for error, 0 for success

# SSAN3-GET-KEYS

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

## Prototype

```
CALL SSAN3-GET-KEYS USING
    BY VALUE      SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE SYSTEM,
    BY REFERENCE POPULATION,
    BY REFERENCE CONTROLS,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY REFERENCE DATA,
    BY REFERENCE COUNT,
    BY REFERENCE KEYS
END CALL
```

## Parameters

SESSION	Refer to the Common Parameters section
SYSTEM	Refer to the Common Parameters section
POPULATION	Refer to the Common Parameters section
CONTROLS	Refer to the Controls section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
DATA	Refer to the REF Controls section
COUNT	A number defining the actual number of keys returned for this name or address
KEYS	An array of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

## Return Code

negative for error, 0 for success

## SSAN3-GET-RANGES

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a `Select` statement to retrieve records from the SSA Keys table.

## Prototype

```
CALL SSAN3-GET-RANGES USING
                                BY VALUE      SOCKH,
                                BY REFERENCE SESSION,
                                BY REFERENCE SYSTEM,
                                BY REFERENCE POPULATION,
                                BY REFERENCE CONTROLS,
                                BY REFERENCE RSP
                                BY REFERENCE SSAMSG,
                                BY REFERENCE DATA,
                                BY REFERENCE COUNT,
                                BY REFERENCE STAB
END CALL
```

## Parameters

SESSION	Refer to the Common Parameters section
SYSTEM	Refer to the Common Parameters section
POPULATION	Refer to the Common Parameters section
CONTROLS	Refer to the Controls section
RSP	Refer to the Common Parameters section

SSAMSG	Refer to the SSA-NAME3 Error Messages section
DATA	Refer to the REF Controls section
COUNT	A number defining the actual number of key ranges returned for this name or address
STAB	An array of pairs of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

### Return Code

negative for error, 0 for success

## SSAN3-INFO

Returns Information regarding the selected System and Population.

### Prototype

```
CALL SSAN3-INFO USING
    BY VALUE      SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE SYSTEM,
    BY REFERENCE POPULATION,
    BY REFERENCE CONTROLS,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY REFERENCE COUNT,
    BY REFERENCE INFO
END CALL
```

### Parameters

SESSION	Refer to the Common Parameters section
SYSTEM	Refer to the Common Parameters section
POPULATION	Refer to the Common Parameters section
CONTROLS	Refer to the Controls section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
COUNT	A number defining the number of rows in the Items Array
INFO	An array of 1024 byte rows, containing the data returned for the specific <code>ssan3_info</code> call

### Return Code

negative for error, 0 for success



## SSAN3-KEYS

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```
CALL SSAN3-KEYS USING
    BY VALUE          SOCKH,
    BY REFERENCE SESSION,
    BY REFERENCE SYSTEM,
    BY REFERENCE POPULATION,
    BY REFERENCE CONTROLS,
    BY REFERENCE RSP ,
    BY REFERENCE SSAMSG,
    BY REFERENCE DATA,
    BY REFERENCE ENCODING,
    BY REFERENCE COUNT,
    BY REFERENCE KEYS ,
    BY VALUE          KEYS-NUM,
    BY VALUE          KEYS-SIZE
END CALL
```

### Parameters

SESSION	Refer to the Common Parameters section
SYSTEM	Refer to the Common Parameters section
POPULATION	Refer to the Common Parameters section
CONTROLS	Refer to the Controls section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
DATA	Refer to the REF Controls section
ENCODING	The encoding used for the data field
COUNT	A number defining the actual number of keys returned for this name or address
KEYS	An array of keys returned for this name or address

### Return Code

negative for error, 0 for success

## SSAN3-MATCH

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

### Prototype

```
CALL SSAN3-MATCH USING
    BY VALUE          SOCKH,
    BY REFERENCE SESSION,
```

```

BY REFERENCE SYSTEM,
BY REFERENCE POPULATION,
BY REFERENCE CONTROLS,
BY REFERENCE RSP ,
BY REFERENCE SSAMSG,
BY REFERENCE SEARCH,
BY REFERENCE FILE ,
BY REFERENCE SCORE,
BY REFERENCE DECISION

```

END CALL

## Parameters

SESSION	Refer to the Common Parameters section
SYSTEM	Refer to the Common Parameters section
POPULATION	Refer to the Common Parameters section
CONTROLS	Refer to the Controls section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the section SSA-NAME3 Error Messages
SEARCH	Refer to the Common Parameters section
FILE	Refer to the Common Parameters section
SCORE	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
DECISION	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

## Return Code

negative for error, 0 for success

## SSAN3-OPEN

An optional function, but its use will improve performance. It opens and initiates an SSA-NAME3 session in preparation.

## Prototype

```

CALL SSAN3-OPEN USING
BY VALUE      SOCKH,
BY REFERENCE SESSION,
BY REFERENCE SYSTEM,
BY REFERENCE POPULATION,
BY REFERENCE CONTROLS,
BY REFERENCE RSP ,
BY REFERENCE SSAMSG
END CALL

```

## Parameters

SESSION	Refer to the Common Parameters section
SYSTEM	Refer to the Common Parameters section
POPULATION	Refer to the Common Parameters section
CONTROLS	Refer to the Controls section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## SSAN3-RANGES

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

## Prototype

```
CALL SSAN3-RANGES USING
                                BY VALUE      SOCKH,
                                BY REFERENCE  SESSION,
                                BY REFERENCE  SYSTEM,
                                BY REFERENCE  POPULATION,
                                BY REFERENCE  CONTROLS,
                                BY REFERENCE  RSP ,
                                BY REFERENCE  SSAMSG,
                                BY REFERENCE  DATA,
                                BY REFERENCE  ENCODING,
                                BY REFERENCE  COUNT,
                                BY REFERENCE  STAB,
                                BY VALUE      STAB-NUM,
                                BY VALUE      STAB-SIZE
END CALL
```

## Parameters

SESSION	Refer to the Common Parameters section
SYSTEM	Refer to the Common Parameters section
POPULATION	Refer to the Common Parameters section
CONTROLS	Refer to the Controls section
RSP	Refer to the Common Parameters section
SSAMSG	Refer to the SSA-NAME3 Error Messages section
DATA	Refer to the REF Controls section

ENCODING	The encoding used for the data field
COUNT	A number defining the actual number of key ranges returned for this name or address
STAB	An array of pairs returned for this name or address

### Return Code

negative for error, 0 for success

## Calling from Java

### Coding the `ssan3_close` Call outside of the `Finalize()` Method

Calling the `ssan3_close` method from the `finalize()` method is not recommended. This is because it delays the call to `ssan3_close` until the Java runtime runs garbage collection on the released object, which can be anytime after the object has been released by the Java program

Instead, it is recommended that the user implement a public cleanup or close method that performs the necessary housekeeping, including a call to `ssan3_close` to release open sessions prior to releasing the object.

For example, calling `ssan3_close` from the `finally` part of a `try/catch/finally` block. Example,

```
static public void main (String arg[]) {
    nm3object w = null;
    try {
        w = new nm3object (arg);
        ....
    } catch (SomeException e) {
        ....
    } finally {
        if (w.h () != -1)
            w.close ();
        if (w != null)
            w.disconnect ();
    }
}
```

### Formatting Search Data and File Data strings

When formatting Search Data and File Data strings, keep in mind that the Java concatenation operator (+) does not scale well. The CPU cycles required rise exponentially as the size and quantity of strings to be concatenated increase. The concatenation operator (+) is only efficient for concatenating a small (2-3) number of short strings. When concatenating a large quantity of strings, the `StringBuffer append()` method will perform better. Once created, a single `StringBuffer` can be reused by setting the string length to 0 with the `setLength(0)` method.

### `ssan3_addr_get_cass_field`

Use this function to retrieve cass specific address fields.

## Prototype

```
using ssa . ssaName3 ;

public synchronized int ssan3_addr_get_cass_field (
    int [ ] session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int suggest_idx , // Long in
    int field_idx , // Long in
    byte [ ] field_value , // Block out
    int field_value_size
) throws SSAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a cass field
field_idx	Specifies a cass field within the nth suggestion
field_value	The cass field value

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_cass\_field\_cnt

Use this function to determine the max number of cass address fields.

## Prototype

```
using ssa.ssaName3 ;

public synchronized int ssan3_addr_get_cass_field_cnt (
    int [ ] session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int [ ] count // Long out
) throws SSAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of cass address fields

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_cass\_field\_info

Use this function to retrieve information about a suggestion.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_get_cass_field_info (
    int [ ] session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int suggest_idx , // Long in
    int [ ] field_length , // LongArray out
    int field_length_num
) throws SSAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each cass address field

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_del\_lines

(deprecated) Use this function to retrieve delivery address line information This function is deprecated. Use **addr\_get\_del\_lines\_ext** instead.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_get_del_lines (
    int [ ] session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int suggest_idx , // Long in
    byte [ ] del_line1 , // Block out
    int del_line1_size ,
    byte [ ] del_line2 , // Block out
    int del_line2_size ,
    byte [ ] del_line3 , // Block out
    int del_line3_size ,
    byte [ ] del_line4 , // Block out
    int del_line4_size ,
    byte [ ] del_line5 , // Block out
    int del_line5_size ,
    byte [ ] del_line6 , // Block out
    int del_line6_size
) throws SSAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string
del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string
del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_del\_lines\_ext

Use this function to retrieve delivery address line information.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_get_del_lines_ext (

    int [ ] session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int suggest_idx , // Long in
    int del_case , // Long in
    byte [ ] del_line1 , // Block out
    int del_line1_size ,
    byte [ ] del_line2 , // Block out
    int del_line2_size ,
        byte [ ] del_line3 , // Block out
        int del_line3_size ,
        byte [ ] del_line4 , // Block out
        int del_line4_size ,
        byte [ ] del_line5 , // Block out
        int del_line5_size ,
        byte [ ] del_line6 , // Block out
        int del_line6_size
) throws SSAAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_case	Specifies delivery address line case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case.
del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string
del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string
del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field

Use this function to retrieve validated address fields.

## Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_get_field (
    int [ ]      session , // Long io
    String [ ]  rsp , // String out
    String [ ]  ssaMsg , // String out
    int        suggest_idx , // Long in
    int        field_idx , // Long in
    byte [ ]   field_value , // Block out
    int        field_value_size ,
    int [ ]    field_val_status , // Long out
    int [ ]    field_val_mods // Long out
) throws SSAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section



suggest_idx	Specifies the nth suggestion from which to get a field
field_idx	Specifies a field within the nth suggestion
field_value	The field value
field_val_status	Specifies how this field matched the validation data
field_val_mods	Specifies how this field was modified by validation data

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_count

Use this function to determine the max number of address fields.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_get_field_count (
    int [ ] session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int [ ] count // Long out
) throws SSAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of address fields

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_ext

Use this function to retrieve all getter fields.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_get_field_ext (
    int [ ] session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int suggest_idx , // Long in
    int field_operation , // Long in
    String field_name , // String in
```

```

        int         field_item_line , // Long in
        String      field_type , // String in
        byte [ ]    field_value , // Block out
        int         field_value_size
    ) throws SSAPIException , SSASocketException ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get fields
field_operation	Field operation Option 0 for AddressElements Option 1 for AddressLines Option 2 for AddressComplete Option 3 for EnrichmentData Option 4 for ResultDataParameter Option 5 for EnrichmentDataStatus
field_name	Refer AD Result.dtd for field names
field_item_line	Represent field line number or field item number
field_type	Refer AD Result.dtd for field attribute Type
field_value	Cleansed field output

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_idx

Use this function to retrieve individual address fields.

### Prototype

```

using ssa.ssaname3 ;

public synchronized int ssan3_addr_get_field_idx (
        int [ ]    session , // Long io
        String [ ] rsp , // String out
        String [ ] ssaMsg , // String out
        int suggest_idx , // Long in
        int field_idx , // Long in
        byte [ ] field_value , // Block out
        int field_value_size
    ) throws SSAPIException , SSASocketException ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

suggest_idx	Specifies the nth suggestion from which to get a field. On successful parse, use 0 for ASM for AD version 4, 1 for ASM for AD version 5
field_idx	Specifies a field within the nth suggestion
field_value	The field value

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_info\_ext

Use this function to retrieve information about a suggestion.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_get_field_info_ext (
    int [ ] session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int suggest_idx , // Long in
    int [ ] field_length , // LongArray out
    int field_length_num ,
    byte [ ] addr_label_encoded , // Block out
    int addr_label_encoded_size ,
    String [ ] addr_label_charset , // String out
    int [ ] score // Long out
) throws SSAAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each address field
addr_label_encoded	The returned label
addr_label_charset	The character set used in the address label
score	The returned label's score

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_len

Use this function to determine the max field length.

## Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_get_field_len (
    int [ ] session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int [ ] max_len // Long out
) throws SSAAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max address field length in bytes

## Return Code

negative for error, 0 for success

# ssan3\_addr\_get\_line\_len

## Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_get_line_len (
    int [ ] session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int [ ] max_len // Long out
) throws SSAAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max line length in bytes

## Return Code

negative for error, 0 for success

# ssan3\_addr\_get\_option

Use this function to set optional parameters.

## Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_get_option (
    int [ ] session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    String param, // String in
    String [ ] value // String out
) throws SSAAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to get
value	Returns the value for the option

## Return Code

negative for error, 0 for success

## ssan3\_addr\_init

Use this function to initialize the Address Standarization interface.

## Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_init (
    int [ ] session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int max_memory // Long in
) throws SSAAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_memory	This field specifies the maximum amount of memory (MB) to be used by the Address Standarization engine

## Return Code

negative for error, 0 for success

## ssan3\_addr\_parse

Use this function to parse an address.

### Prototype

```
using ssa.ssaname3 ;
public synchronized int ssan3_addr_parse (
    int [ ]          session , // Long io
    String [ ]      rsp , // String out
    String [ ]      ssaMsg , // String out
    int [ ]         field_length , // LongArray out
    int field_length_num
) throws SSAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_length	An array containing the length of each parsed field

### Return Code

negative for error, 0 for success

## ssan3\_addr\_preload\_country

Use this function to preload country database.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_preload_country (
    int [ ]          session , // Long io
    String [ ]      rsp , // String out
    String [ ]      ssaMsg , // String out
    String          preload_type , // String in
    String          preload_country , // String in
    String          val_mode // String in
) throws SSAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
preload_type	Type of preload to perform
preload_country	Country database to be preloaded
val_mode	This field specified validation mode

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_attrib

Use this function to specify the character set of the data and a default country.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_set_attrib (
    int [ ]    session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    String     char_set , // String in
    String     default_country // String in
) throws SSAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
char_set	The name of the character set used to encode the input and output
default_country	The default country used for validation when parsing cannot detect a country name

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_del\_lines

Use this function to set delivery address line information.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_set_del_lines (
    int [ ]    session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    byte [ ]   del_line1 , // Block in
    byte [ ]   del_line2 , // Block in
    byte [ ]   del_line3 , // Block in
    byte [ ]   del_line4 , // Block in
    byte [ ]   del_line5 , // Block in
    byte [ ]   del_line6 // Block in
) throws SSAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
del_line1	Delivery address line 1 input string
del_line2	Delivery address line 2 input string
del_line3	Delivery address line 3 input string
del_line4	Delivery address line 4 input string
del_line5	Delivery address line 5 input string
del_line6	Delivery address line 6 input string

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_field\_case

Use this function to set individual input fields case option.

## Prototype

```
using ssa.ssaName3 ;

public synchronized int ssan3_addr_set_field_case (
    int [ ] session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int field_idx , // Long in
    int field_case // Long
) throws SSAAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_case	Specifies output field case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case

## Return Code

negative for error, 0 for success



## ssan3\_addr\_set\_field\_idx

Use this function to set individual input fields by idx.

### Prototype

```
using ssa.  
ssaname3 ;  
public synchronized int ssan3_addr_set_field_idx (  
    int [ ] session , // Long io  
    String [ ] rsp , // String out  
    String [ ] ssaMsg , // String out  
    int field_idx , // Long in  
    byte [ ] field_value // Block in  
    ) throws SSAAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_value	Specifies a value for the nth field

### Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_field\_name

Use this function to set individual input fields by name.

### Prototype

```
using ssa.ssaname3 ;  
  
public synchronized int ssan3_addr_set_field_name (  
    int [ ] session , // Long io  
    String [ ] rsp , // String out  
    String [ ] ssaMsg , // String out  
    String field_name , // String in  
    byte [ ] field_value // Block in  
    ) throws SSAAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_name	Specifies the name of the field to set
field_value	Specifies a value for the field

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_lines

Use this function to provide an address to parse or validate.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_set_lines (
    int [ ]    session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    byte [ ]   line_1 , // Block in
    byte [ ]   line_2 , // Block in
    byte [ ]   line_3 , // Block in
    byte [ ]   line_4 , // Block in
    byte [ ]   line_5 , // Block in
    byte [ ]   line_6 , // Block in
    byte [ ]   line_7 , // Block in
    byte [ ]   line_8 , // Block in
    byte [ ]   line_9 , // Block in
    byte [ ]   line_10 // Block in
) throws SSAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
line_1	The first line of the address
line_2	The second line of the address
line_3	The third line of the address
line_4	The fourth line of the address
line_5	The fifth line of the address
line_6	The sixth line of the address
line_7	The seventh line of the address
line_8	The eighth line of the address
line_9	The ninth line of the address
line_10	The tenth line of the address

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_option

Use this function to set optional parameters.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_set_option (
    int [ ]    session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    String     param , // String in
    String     value  // String in
) throws SSAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to set
value	This field specifies a value for the option

### Return Code

negative for error, 0 for success

## ssan3\_addr\_validate

Use this function to validate an address.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_addr_validate (
    int [ ]    session , // Long io
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int [ ]    status , // Long out
    int [ ]    n_suggest // Long out
) throws SSAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
status	The status returned by the validation process
n_suggest	The number of suggestions generated by validation

## Return Code

negative for error, 0 for success

## ssan3\_close

Closes the SSA session and deallocates memory. Similarly to **ssa\_open**, it is optional.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_close (
    int [ ] session , // Long io
    String    system , // String in
    String    population , // String in
    String    controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg // String out
) throws SSAAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## ssan3\_convert\_keys

Used to convert 8 byte keys (from server) to 5 byte keys (for client).

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_convert_keys (
    String [ ] keys8 , // StringArray in
    byte [ ] [ ] keys5 , // BlockArray out
    int [ ] keys5_num ,
    int [ ] keys5_size ,
    String [ ] rsp , // String out
    String [ ] ssaMsg // String out
) throws SSAAPIException , SSASocketException ;
```

## Parameters

keys8	Block of 8 byte keys to be converted
keys5	Block of 5 byte keys (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## ssan3\_convert\_ranges

Used to convert 8 byte ranges (from server) to 5 byte ranges (for client).

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_convert_ranges (
    String [ ] ranges8 , // StringArray in
    byte [ ] [ ] ranges5 , // BlockArray out
    int [ ] ranges5_num ,
    int [ ] ranges5_size ,
    String [ ] rsp , // String out
    String [ ] ssaMsg // String out
) throws SSAAPIException , SSASocketException ;
```

## Parameters

ranges8	Block of 8 byte ranges to be converted
ranges5	Block of 5 byte ranges (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## ssan3\_disconnect

Releases resources allocated to a socket.

### Prototype

```
using ssa.ssaname3 ;
public synchronized int ssan3_disconnect ( ) throws SSAAPIException ,
SSASocketException ;
```

## Parameters

none

## Return Code

negative for error, 0 for success

## ssan3\_errors\_get\_all

Get the Server side error messages from the last API function that failed. This function should be called repeatedly until it returns 1, meaning all messages have been retrieved.

**Note:** If a communication (socket) error occurred, this function will also fail. Refer to the *OPERATIONS MANUAL*, *Error Log* section for information on interpreting the Error Log.

## Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_errors_get_all (
    String [ ] msg // String out
) throws SSAAPIException , SSASocketException ;
```

## Parameters

msg	This is an error message
-----	--------------------------

## Return Code

negative for error, 0 for success

## ssan3\_get\_keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

## Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_get_keys (
    int [ ] session , // Long io
    String system , // String in
    String population , // String in
    String controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    String data , // String in
    int [ ] count , // Long out
    String [ ] keys // StringArray out
) throws SSAAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section

rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

### Return Code

negative for error, 0 for success

## ssan3\_get\_keys\_encoded

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_get_keys_encoded (
    int [ ] session , // Long io
    String system , // String in
    String population , // String in
    String controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    String data , // EncodedString in
    String dataEncType ,
    int [ ] count , // Long out
    String [ ] keys // StringArray out
) throws SSAAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

## Return Code

negative for error, 0 for success

## ssan3\_get\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_get_ranges (
    int [ ] session , // Long io
    String system , // String in
    String population , // String in
    String controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    String data , // String in
    int [ ] count , // Long out
    String [ ] stab // StringArray out
) throws SSAAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or address

## Return Code

negative for error, 0 for success

## ssan3\_get\_ranges\_encoded

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```
using ssa.ssaname3 ;
```



```

public synchronized int ssan3_get_ranges_encoded (
    int [] session , // Long io
    String system , // String in
    String population , // String in
    String controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    String data , // EncodedString in
    String dataEncType ,
    int [ ] count , // Long out
    String [ ] stab // StringArray out
) throws SSAAPIException , SSASocketException ;

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or address

## Return Code

negative for error, 0 for success

## ssan3\_info

Returns Information regarding the selected System and Population.

### Prototype

```

using ssa.ssaname3 ;

public synchronized int ssan3_info (
    int [] session , // Long io
    String system , // String in
    String population , // String in
    String controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int [ ] count , // Long out
    String [ ] info // StringArray out
) throws SSAAPIException , SSASocketException ;

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the section Controls
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	A number defining the number of rows in the Items Array
info	An array of 1024 byte rows, containing the data returned for the specific <b>ssan3_info</b> call

## Return Code

negative for error, 0 for success

## ssan3\_info\_ext

Returns information about the selected system and population for the following **ssan3\_info** control items:

- map\_profile
- map\_population
- map\_data
- map\_field

## Prototype

```
using ssa.ssaName3 ;

public synchronized int ssan3_info_ext (
    int [ ] session , // Long in
    String system_name , // String in
    String population , // String in
    String controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int in_count , // Long in
    String [ ] info_count , // StringArray in
    int [ ] count , // Long out
    String [ ] info // StringArray out
    int infoSize // StringArray out
) throws SSAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section.
system	Refer to the Common Parameters section.
population	Refer to the Common Parameters section.
controls	Refer to the section Controls.

ssaMsg	Refer to the SSA-NAME3 Error Messages section.
count	A number defining the number of rows in the Items Array.
in_count	Number of records that the in_info parameter returns.
in_info	Array of records for the following ssan3_info control items: - map_profile - map_population - map_data - map_field
info	An array of 1024 byte rows, containing the data returned for the specific <b>ssan3_info</b> call.

### Return Code

Negative for error and 0 for success.

## ssan3\_keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```

using ssa.ssaname3 ;

public synchronized int ssan3_keys (
    int [ ] session , // Long in
    String system , // String in
    String population , // String in
    String controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    byte [ ] data , // Block in
    String encoding , // String in
    int [ ] count , // Long out
    byte [ ] [ ] keys , // BlockArray out
    int [ ] keys_num,
    int [ ] keys_size
) throws SSAAPIException , SSASocketException ;

```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section

encoding	The encoding used for the data field
count	A number defining the actual number of keys returned for this name or address
keys	An array of keys returned for this name or address

### Return Code

negative for error, 0 for success

## ssan3\_match

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_match (
    int [ ] session , // Long io
    String    system , // String in
    String    population , // String in
    String    controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    String    search , // String in
    String file , // String in
    String [ ] score , // String out
    String [ ] decision // String out
) throws SSAAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section
file	Refer to the Common Parameters section

score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

### Return Code

negative for error, 0 for success

## ssan3\_match\_encoded

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_match_encoded (
    int [ ] session , // Long io
    String system , // String in
    String population , // String in
    String controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    String search , // EncodedString in
    String searchEncType ,
    String file , // EncodedString in
    String fileEncType ,
    String [ ] score , // String out
    String [ ] decision // String out
) throws SSAAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section
file	Refer to the Common Parameters section
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

## Return Code

negative for error, 0 for success

## ssan3\_match\_ext

Returns a score, match decision, and match report for the search and file records. After the candidate records are identified based on the ranges that the **ssan3\_get\_ranges\_ext** call returns, the **ssan3\_match\_ext** call qualifies the candidate records.

## Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_match (
    int [ ] session , // Long in
    String  system_name , // String in
    String  population , // String in
    String  controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    String search , // String in
    int s_len , // Long in
    String s_encoding , // String in
    String file , // String in
    int f_len , // Long in
    String f_encoding , // String in
    String [ ] score , // String out
    String [ ] decision // String out
    int [ ] count , // Long out
    String [ ] explain_data , // StringArray out
    int explain_data_size , // StringArray out
) throws SSAException;
```

## Parameters

session	Refer to the Common Parameters section.
system	Refer to the Common Parameters section.
population	Refer to the Common Parameters section.
controls	Refer to the Controls section.
rsp	Refer to the Common Parameters section.
ssaMsg	Refer to the SSA-NAME3 Error Messages section.
search	Refer to the Common Parameters section.
file	Refer to the Common Parameters section.
score	A 3-byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, match purpose and match level.

decision	'A' meaning 'Accept'. The score is greater than or equal to the accept limit for the specified match purpose and match level. 'U' meaning 'Undecided'. The score is between the accept limit and reject limit for the specified match purpose and match level. 'R' meaning 'Reject'. The score is less than or equal to the reject limit for the specified match purpose and match level.
explain_data	Returns the match report.

### Return Code

Negative for error and 0 for success.

## ssan3\_match\_rescore

Returns the score and match decision based on the revised field weights.

### Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_match (
    int [ ] session , // Long io
    String  system_name , // String in
    String  population , // String in
    String  controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    int count , // Long in
    String [ ] explain_data , // StringArray in
    String [ ] score , // String out
    String [ ] decision // String out
) throws SSAException;
```

### Parameters

session	Refer to the Common Parameters section.
system	Refer to the Common Parameters section.
population	Refer to the Common Parameters section.
controls	Refer to the Controls section.
rsp	Refer to the Common Parameters section.
ssaMsg	Refer to the SSA-NAME3 Error Messages section.
explain_data	Returns the match report.

score	'A' meaning 'Accept'. The score is greater than or equal to the accept limit for the specified match purpose and match level. 'U' meaning 'Undecided'. The score is between the accept limit and reject limit for the specified match purpose and match level. 'R' meaning 'Reject'. The score is less than or equal to the reject limit for the specified match purpose and match level.
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level.

### Return Code

Negative for error and 0 for success.

## ssan3\_open

An optional function, but its use will improve performance. It opens and initiates an SSA-NAME3 session in preparation.

### Prototype

```
using ssa.ssaName3 ;

public synchronized int ssan3_open (
    int [ ] session , // Long io
    String system , // String in
    String population , // String in
    String controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg // String out
) throws SSAAPIException , SSASocketException ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## ssan3\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.



## Prototype

```
using ssa.ssaname3 ;

public synchronized int ssan3_ranges (
    int [ ] session , // Long io
    String system , // String in
    String population , // String in
    String controls , // String in
    String [ ] rsp , // String out
    String [ ] ssaMsg , // String out
    byte [ ] data , // Block in
    String encoding , // String in
    int [ ] count , // Long out
    byte [ ] [ ] stab , // BlockArray out
    int [ ] stab_num ,
    int [ ] stab_size
) throws SSAAPIException , SSASocketException ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs returned for this name or address

## Return Code

negative for error, 0 for success

# Calling from Microsoft SQL Server

## Prerequisites

Before you access SSA-NAME3 functionality from Microsoft SQL Server, perform the following prerequisite tasks:

1. Copy the <Informatica Installation Directory>\mssql\ssan3xp.dll file to the <Microsoft SQL Server Installation Directory>\Binn directory.

2. Copy the following files from the <Informatica Installation Directory>\bin directory to the <Microsoft SQL Server Installation Directory>\Binn directory:

- ssan3cl.dll
- ssaiok.dll
- ssali.dll
- ssan3v2.dll
- ssan3tb.dll

3. Create the following system environment variables:

- SSA\_LIHOST=<License Server Host>:<Port>
- SSAPR=<Informatica Installation Directory>\pr

For information about creating environment variables on Windows, consult the Windows documentation.

4. Restart the **SQL Server (MSSQLSERVER)** service.

## ssan3xp\_addr\_get\_cass\_field

Use this function to retrieve cass specific address fields.

### Prototype

```
EXEC master.dbo.ssan3xp_addr_get_cass_field (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @suggest_idx , - Long in
    @field_idx , - Long in
    @field_value OUTPUT,- Block out
    @field_value_size
);
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a cass field
field_idx	Specifies a cass field within the nth suggestion
field_value	The cass field value

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_get\_cass\_field\_cnt

Use this function to determine the max number of cass address fields.

## Prototype

```
EXEC master.dbo.ssan3xp_addr_get_cass_field_cnt (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @count OUTPUT - Long out
);
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of cass address fields

## Return Code

negative for error, 0 for success

## ssan3xp\_addr\_get\_cass\_field\_info

Use this function to retrieve information about a suggestion.

## Prototype

```
CREATE TABLE #ssan3xp_addr_get_cass_field_info_layout (
    field_length INT)
INSERT #ssan3xp_addr_get_cass_field_info_layout
EXEC
master.dbo.ssan3xp_addr_get_cass_field_info_layout
@rc OUTPUT,
@sockh ,
@session OUTPUT, - Long io
@rsp OUTPUT, - String out
@ssaMsg OUTPUT, - String out
@suggest_idx , - Long in
@field_length OUTPUT,- LongArray out
@field_length_num ;
if @rc < 0 begin
    goto ret ;
end ;
DECLARE ssan3xp_addr_get_cass_field_info_cursor CURSOR SCROLL FOR
SELECT
    field_length
FROM #ssan3xp_addr_get_cass_field_info
OPEN ssan3xp_addr_get_cass_field_info_cursor ;
FETCH FIRST FROM ssan3xp_addr_get_cass_field_info_cursor INTO
    field_length ;
WHILE @@FETCH_STATUS = 0 begin
    . . .
END;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each cass address field

## Return Code

negative for error, 0 for success

## ssan3xp\_addr\_get\_del\_lines

(deprecated) Use this function to retrieve delivery address line information This function is deprecated. Use **addr\_get\_del\_lines\_ext** instead.

## Prototype

```
EXEC master.dbo.ssan3xp_addr_get_del_lines (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @suggest_idx , - Long in
    @del_line1 OUTPUT,- Block out
    @del_line1_size ,
    @del_line2 OUTPUT,- Block out
    @del_line2_size ,
    @del_line3 OUTPUT,- Block out
    @del_line3_size ,
    @del_line4 OUTPUT,- Block out
    @del_line4_size ,
    @del_line5 OUTPUT,- Block out
    @del_line5_size ,
    @del_line6 OUTPUT,- Block out
    @del_line6_size
);
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string

del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string
del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_get\_del\_lines\_ext

Use this function to retrieve delivery address line information.

### Prototype

```
EXEC master.dbo.ssan3xp_addr_get_del_lines_ext (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @suggest_idx , - Long in
    @del_case , - Long in
    @del_line1 OUTPUT,- Block out
    @del_line1_size ,
    @del_line2 OUTPUT,- Block out
    @del_line2_size ,
    @del_line3 OUTPUT,- Block out
    @del_line3_size ,
    @del_line4 OUTPUT,- Block out
    @del_line4_size ,
    @del_line5 OUTPUT,- Block out
    @del_line5_size ,
    @del_line6 OUTPUT,- Block out
    @del_line6_size
);
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_case	Specifies delivery address line case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case
del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string

del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string
del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_get\_field

Use this function to retrieve validated address fields.

### Prototype

```
EXEC master.dbo.ssan3xp_addr_get_field (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @suggest_idx , - Long in
    @field_idx , - Long in
    @field_value OUTPUT,- Block out
    @field_value_size ,
    @field_val_status OUTPUT,- Long out
    @field_val_mods OUTPUT - Long out
);
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a field
field_idx	Specifies a field within the nth suggestion
field_value	The field value
field_val_status	Specifies how this field matched the validation data
field_val_mods	Specifies how this field was modified by validation data

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_get\_field\_count

Use this function to determine the max number of address fields.

## Prototype

```
EXEC master.dbo.ssan3xp_addr_get_field_count (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @count OUTPUT - Long out
);
```

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of address fields

## Return Code

negative for error, 0 for success

## ssan3xp\_addr\_get\_field\_ext

Use this function to retrieve all getter fields.

## Prototype

```
EXEC master.dbo.ssan3xp_addr_get_field_ext (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @suggest_idx , - Long in
    @field_operation , - Long in
    @field_name , - String in
    @field_item_line , - Long in
    @field_type , - String in
    @field_value OUTPUT, - Block out
    @field_value_size
);
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get fields
field_operation	Field operation Option 0 for AddressElements Option 1 for AddressLines Option 2 for AddressComplete Option 3 for EnrichmentData Option 4 for ResultDataParameter Option 5 for EnrichmentDataStatus
field_name	Refer AD Result.dtd for field names

field_item_line	Represent field line number or field item number
field_type	Refer AD Result.dtd for field attribute Type
field_value	Cleansed field output

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_get\_field\_idx

Use this function to retrieve individual address fields.

### Prototype

```
EXEC master.dbo.ssan3xp_addr_get_f field_idx (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @suggest_idx , - Long in
    @field_idx , - Long in
    @field_value OUTPUT,- Block out
    @field_value_size
);
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a field. On successful parse, use 0 for ASM for AD version 4, 1 for ASM for AD version 5
field_idx	Specifies a field within the nth suggestion
field_value	The field value

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_get\_field\_info\_ext

Use this function to retrieve information about a suggestion.

### Prototype

```
CREATE TABLE #ssan3xp_addr_get_field_info_ext_layout (
    field_length INT)
INSERT #ssan3xp_addr_get_field_info_ext_layout
EXEC master.dbo.ssan3xp_addr_get_field_info_ext_layout
```



```

@rc OUTPUT,
@sockh ,
@session OUTPUT, - Long i o
@rsp OUTPUT, - String out
@ssaMsg OUTPUT, - String out
@suggest_idx , - Long in
@field_length OUTPUT,-

LongArray out

Block out

OUTPUT,- String out

if @rc < 0 begin
    goto ret ;
end ;
DECLARE ssan3xp_addr_get_field_info_ext_cursor CURSOR SCROLL FOR
SELECT
    field_length
FROM #ssan3xp_addr_get_field_info_ext
OPEN ssan3xp_addr_get_field_info_ext_cursor ;
FETCH FIRST FROM ssan3xp_addr_get_field_info_ext_cursor INTO
    field_length ;
WHILE @@FETCH_STATUS = 0 begin
    .
    .
    .
END;

@field_length_num ,
@addr_label_encoded OUTPUT,-

@addr_label_encoded_size ,
@addr_label_charset

@score OUTPUT; - Long out

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each address field
addr_label_encoded	The returned label
addr_label_charset	The character set used in the address label
score	The returned label's score

## Return Code

negative for error, 0 for success

## ssan3xp\_addr\_get\_field\_len

Use this function to determine the max field length.

### Prototype

```

EXEC master.dbo.ssan3xp_addr_get_field_len (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out

```

```

        @max_len OUTPUT - Long out
    ) ;

```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max address field length in bytes

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_get\_line\_len

### Prototype

```

EXEC master.dbo.ssan3xp_addr_get_line_len (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @max_len OUTPUT - Long out
) ;

```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max line length in bytes

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_get\_option

Use this function to set optional parameters.

### Prototype

```

EXEC master.dbo.ssan3xp_addr_get_option (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @param, - String in
) ;

```

```

        @value OUTPUT - String out
    ) ;

```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to get
value	Returns the value for the option

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_init

Use this function to initialize the Address Standardization interface.

### Prototype

```

EXEC master.dbo.ssan3xp_addr_init (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @max_memory - Long in
) ;

```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_memory	This field specifies the maximum amount of memory (MB) to be used by the Address Standardization engine

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_parse

Use this function to parse an address.

### Prototype

```

CREATE TABLE #ssan3xp_addr_parse_layout (
    field_length INT)
INSERT #ssan3xp_addr_parse_layout

```

```

EXEC master.dbo.ssan3xp_addr_parse_layout
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, .... Long io
    @rsp OUTPUT, .... String out
    @ssaMsg OUTPUT, .... String out
    @field_length OUTPUT,.... LongArray out
    @field_length_num ;
if @rc < 0 begin
    goto ret ;
end ;
DECLARE ssan3xp_addr_parse_cursor CURSOR SCROLL FOR
SELECT
    field_length
FROM #ssan3xp_addr_parse
OPEN ssan3xp_addr_parse_cursor ;
FETCH FIRST FROM ssan3xp_addr_parse_cursor INTO
    field_length ;
WHILE @@FETCH_STATUS = 0 begin
    . . .
END;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_length	An array containing the length of each parsed field

## Return Code

negative for error, 0 for success

# ssan3xp\_addr\_preload\_country

Use this function to preload country database.

## Prototype

```

EXEC master.dbo.ssan3xp_addr_preload_country (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @preload_type , - String in
    @preload_country ,- String in
    @val_mode - String in
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

preload_type	Type of preload to perform
preload_country	Country database to be preloaded
val_mode	This field specified validation mode

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_set\_attrb

Use this function to specify the character set of the data and a default country.

### Prototype

```
EXEC master.dbo.ssan3xp_addr_set_attrb (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @char_set , - String in
    @default_country - String in
);
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
char_set	The name of the character set used to encode the input and output
default_country	The default country used for validation when parsing cannot detect a country name

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_set\_del\_lines

Use this function to set delivery address line information.

### Prototype

```
EXEC master.dbo.ssan3xp_addr_set_del_lines (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @del_line1 , - Block in
    @del_line2 , - Block in
    @del_line3 , - Block in
    @del_line4 , - Block in
);
```

```

        @del_line5 , - Block in
        @del_line6 - Block in
    ) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
del_line1	Delivery address line 1 input string
del_line2	Delivery address line 2 input string
del_line3	Delivery address line 3 input string
del_line4	Delivery address line 4 input string
del_line5	Delivery address line 5 input string
del_line6	Delivery address line 6 input string

## Return Code

negative for error, 0 for success

# ssan3xp\_addr\_set\_field\_case

Use this function to set individual input fields case option.

## Prototype

```

EXEC master.dbo.ssan3xp_addr_set_field_case (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @field_idx , - Long in
    @field_case - Long in
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_case	Specifies output field case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case

## Return Code

negative for error, 0 for success

## ssan3xp\_addr\_set\_field\_idx

Use this function to set individual input fields by idx.

### Prototype

```
EXEC master.dbo.ssan3xp_addr_set_field_idx (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @field_idx , - Long in
    @field_value - Block in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_value	Specifies a value for the nth field

## Return Code

negative for error, 0 for success

## ssan3xp\_addr\_set\_field\_name

Use this function to set individual input fields by name.

### Prototype

```
EXEC master.dbo.ssan3xp_addr_set_field_name (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @field_name , - String in
    @field_value - Block in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section

ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_name	Specifies the name of the field to set
field_value	Specifies a value for the field

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_set\_lines

Use this function to provide an address to parse or validate.

### Prototype

```
EXEC master.dbo.ssan3xp_addr_set_lines (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @line_1 , - Block in
    @line_2 , - Block in
    @line_3 , - Block in
    @line_4 , - Block in
    @line_5 , - Block in
    @line_6 , - Block in
    @line_7 , - Block in
    @line_8 , - Block in
    @line_9 , - Block in
    @line_10 - Block in
);
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
line_1	The first line of the address
line_2	The second line of the address
line_3	The third line of the address
line_4	The fourth line of the address
line_5	The fifth line of the address
line_6	The sixth line of the address
line_7	The seventh line of the address
line_8	The eighth line of the address



line_9	The ninth line of the address
line_10	The tenth line of the address

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_set\_option

Use this function to set optional parameters.

### Prototype

```
EXEC master.dbo.ssan3xp_addr_set_option (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @param, - String in
    @value - String in
);
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to set
value	This field specifies a value for the option

### Return Code

negative for error, 0 for success

## ssan3xp\_addr\_validate

Use this function to validate an address.

### Prototype

```
EXEC master.dbo.ssan3xp_addr_validate (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @status OUTPUT, - Long out
    @n_suggest OUTPUT - Long out
);
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
status	The status returned by the validation process
n_suggest	The number of suggestions generated by validation

## Return Code

negative for error, 0 for success

## ssan3xp\_close

Closes the SSA session and deallocates memory. Similarly to **ssa\_open**, it is optional.

## Prototype

```
EXEC master.dbo.ssan3xp_close (  
    @rc OUTPUT,  
    @sockh ,  
    @session OUTPUT, - Long io  
    @system , - String in  
    @population , - String in  
    @controls , - String in  
    @rsp OUTPUT, - String out  
    @ssaMsg OUTPUT - String out  
);
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## ssan3xp\_connect

Initiates a socket.

## Prototype

```
EXEC master.dbo.ssan3xp_connect (
    @rc OUTPUT,
    @host , - String in
    @port , - Long in
    @sockh OUTPUT - Long out
) ;
```

## Parameters

host	This is the host to connect to
port	This is the port to connect to
sockh	This is a socket handle

## Return Code

negative for error, 0 for success

## ssan3xp\_convert\_keys

Used to convert 8 byte keys (from server) to 5 byte keys (for client).

## Prototype

```
CREATE TABLE #ssan3xp_convert_keys_layout (
    keys5 VARCHAR(256))
INSERT #ssan3xp_convert_keys_layout
EXEC master.dbo.ssan3xp_convert_keys_layout
    @rc OUTPUT,
    @sockh ,
    @keys8 , - StringArray in
    @keys5 OUTPUT, - BlockArray out
    @keys5_num,
    @keys5_size ,
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT; - String out
if @rc < 0 begin
    goto ret ;
end ;
DECLARE ssan3xp_convert_keys_cursor CURSOR SCROLL FOR
SELECT
    keys5
FROM #ssan3xp_convert_keys
OPEN ssan3xp_convert_keys_cursor ;
FETCH FIRST FROM ssan3xp_convert_keys_cursor INTO
    keys5 ;
WHILE @@FETCH_STATUS = 0 begin
    . . .
END;
```

## Parameters

keys8	Block of 8 byte keys to be converted
keys5	Block of 5 byte keys (output)

rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## ssan3xp\_convert\_ranges

Used to convert 8 byte ranges (from server) to 5 byte ranges (for client).

### Prototype

```

CREATE TABLE #ssan3xp_convert_ranges_layout (
ranges5 VARCHAR(256) )
INSERT #ssan3xp_convert_ranges_layout
EXEC master.dbo.ssan3xp_convert_ranges_layout
@rc OUTPUT,
@sockh ,
@ranges8 , - StringArray in
@ranges5 OUTPUT, - BlockArray out
@ranges5_num ,
@ranges5_size ,
@rsp OUTPUT, - String out
@ssaMsg OUTPUT; - String out
if @rc < 0 begin
goto r e t ;
end ;
DECLARE ssan3xp_convert_ranges_cursor CURSOR SCROLL FOR
SELECT
ranges5
FROM #ssan3xp_convert_ranges
OPEN ssan3xp_convert_ranges_cursor ;
FETCH FIRST FROM ssan3xp_convert_ranges_cursor INTO
ranges5 ;
WHILE @@FETCH_STATUS = 0 begin
. . .
END;

```

### Parameters

ranges8	Block of 8 byte ranges to be converted
ranges5	Block of 5 byte ranges (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## ssan3xp\_disconnect

Releases resources allocated to a socket.

## Prototype

```
EXEC master.dbo.ssan3xp_disconnect (
                                @rc OUTPUT,
                                @sockh
) ;
```

## Parameters

none

## Return Code

negative for error, 0 for success

## ssan3xp\_errors\_get\_all

Get the Server side error messages from the last API function that failed. This function should be called repeatedly until it returns 1, meaning all messages have been retrieved.

**Note:** If a communication (socket) error occurred, this function will also fail. Refer to the *OPERATIONS MANUAL, Error Log* section for information on interpreting the Error Log.

## Prototype

```
EXEC master.dbo.ssan3xp_errors_get_all (
                                @rc OUTPUT,
                                @sockh ,
                                @msg OUTPUT - String out
) ;
```

## Parameters

msg is an error message.

## Return Code

negative for error, 0 for success

## get\_keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

## Prototype

```
using ssa;

my @keys = $ssa->get_keys (
    $controls , # String in
    $data      # String in
) ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section

controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or address

### Return Code

negative for error, 0 for success

## ssan3xp\_get\_keys\_encoded\_text

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```

CREATE TABLE #ssan3xp_get_keys_encoded_text_layout (
    keys VARCHAR(256))
INSERT #ssan3xp_get_keys_encoded_text_layout
EXEC master.dbo.ssan3xp_get_keys_encoded_text_layout
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @system , - String in
    @population , - String in
    @controls , - String in
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @data , - EncodedString in
    @count OUTPUT, - Long out
    @keys OUTPUT; - StringArray out
if @rc < 0 begin
    goto ret ;
end ;
DECLARE ssan3xp_get_keys_encoded_text_cursor CURSOR SCROLL FOR
SELECT
    keys
FROM #ssan3xp_get_keys_encoded_text
OPEN ssan3xp_get_keys_encoded_text_cur sor ;
FETCH FIRST FROM ssan3xp_get_keys_encoded_text_cursor INTO
    keys ;
WHILE @@FETCH_STATUS = 0 begin
    . . .
END;

```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section

controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

### Return Code

negative for error, 0 for success

## ssan3xp\_get\_keys\_encoded\_unicode

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```

CREATE TABLE #ssan3xp_get_keys_encoded_unicode_layout (
    keys      VARCHAR(256))
INSERT #ssan3xp_get_keys_encoded_unicode_layout
EXEC master.dbo.ssan3xp_get_keys_encoded_unicode_layout
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @system , - String in
    @population , - String in
    @controls , - String in
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @data , - EncodedString in
    @count OUTPUT, - Long out
    @keys OUTPUT; - StringArray out
if @rc < 0 begin
    goto ret ;
end ;
DECLARE ssan3xp_get_keys_encoded_unicode_cursor CURSOR SCROLL FOR
SELECT
    keys
FROM #ssan3xp_get_keys_encoded_unicode
OPEN ssan3xp_get_keys_encoded_unicode_cursor ;
FETCH FIRST FROM ssan3xp_get_keys_encoded_unicode_cursor INTO
    keys ;
WHILE @@FETCH_STATUS = 0 begin
    . . .
END;

```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section

controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

### Return Code

negative for error, 0 for success

## ssan3xp\_get\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```

CREATE TABLE #ssan3xp_get_ranges_layout (
    stab VARCHAR(256))
INSERT #ssan3xp_get_ranges_layout
EXEC master.dbo.ssan3xp_get_ranges_layout
@rc OUTPUT,
@sockh ,
@session OUTPUT, - Long i o
@system , - String in
@population , - String in
@controls , - String in
@rsp OUTPUT, - String out
@ssaMsg OUTPUT, - String out
@data , - EncodedString in
@count OUTPUT, - Long out
@stab OUTPUT; - StringArray out
if @rc < 0 begin
    goto r e t ;
end ;
DECLARE ssan3xp_get_ranges_cursor CURSOR SCROLL FOR
SELECT
    stab
FROM # ssan3xp_get_ranges
OPEN ssan3xp_get_ranges_cursor ;
FETCH FIRST FROM ssan3xp_get_ranges_cursor INTO
    stab ;
WHILE @@FETCH_STATUS = 0 begin
    . . .
END;

```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section



controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

### Return Code

negative for error, 0 for success

## get\_ranges\_encoded

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```
using ssa;

my @stab = $ssa->get_ranges_encoded (
    $controls ,      # String in
    $data ,          # EncodedString in
    $dataEncType
);
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

### Return Code

negative for error, 0 for success

## ssan3xp\_get\_ranges\_encoded\_unicode

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```
CREATE TABLE #ssan3xp_get_ranges_encoded_unicode_layout (
    stab VARCHAR(256))
INSERT #ssan3xp_get_ranges_encoded_unicode_layout
EXEC master.dbo.ssan3xp_get_ranges_encoded_unicode_layout
@rc OUTPUT,
@sockh ,
@session OUTPUT, - Long io
@system , - String in
@population , - String in
@controls , - String in
@rsp OUTPUT, - String out
@ssaMsg OUTPUT, - String out
@data , - EncodedString in
@count OUTPUT, - Long out
@stab OUTPUT; - StringArray out

if @rc < 0 begin
    goto ret ;
end ;
DECLARE ssan3xp_get_ranges_encoded_unicode_cursor CURSOR SCROLL FOR
SELECT
    stab
FROM #ssan3xp_get_ranges_encoded_unicode
OPEN ssan3xp_get_ranges_encoded_unicode_cursor ;
FETCH FIRST FROM ssan3xp_get_ranges_encoded_unicode_cursor INTO
    stab ;
WHILE @@FETCH_STATUS = 0 begin
    . . .
END;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or addressKEY_SIZE

### Return Code

negative for error, 0 for success

## ssan3xp\_info

Returns Information regarding the selected System and Population.

### Prototype

```
CREATE TABLE #ssan3xp_info_layout (
    info VARCHAR(256))
INSERT #ssan3xp_info_layout
EXEC master.dbo.ssan3xp_info_layout
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @system , - String in
    @population , - String in
    @controls , - String in
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @count OUTPUT, - Long out
    @info OUTPUT; - StringArray out

if @rc < 0 begin
    goto ret ;
end ;
DECLARE ssan3xp_info_cursor CURSOR SCROLL FOR
SELECT
    info
FROM #ssan3xp_info
OPEN ssan3xp_info_cursor ;
FETCH FIRST FROM ssan3xp_info_cursor INTO
    info ;
WHILE @@FETCH_STATUS = 0 begin
    . . .
END;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	A number defining the number of rows in the Items Array
info	An array of 1024 byte rows, containing the data returned for the specific <b>ssan3_info</b> call

### Return Code

negative for error, 0 for success

## ssan3xp\_keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

## Prototype

```
CREATE TABLE #ssan3xp_keys_layout (
    keys    VARCHAR(256))

INSERT #ssan3xp_keys_layout
EXEC master.dbo.ssan3xp_keys_layout
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @system , - String in
    @population , - String in
    @controls , - String in
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @data , - Block in
    @encoding , - String in
    @count OUTPUT, - Long out
    @keys OUTPUT, - BlockArray out
    @keys_num,
    @keys_size ;

if @rc < 0 begin
    goto ret ;
end ;
DECLARE ssan3xp_keys_cursor CURSOR SCROLL FOR
SELECT
    keys
FROM # ssan3xp_keys
OPEN ssan3xp_keys_cursor ;
FETCH FIRST FROM ssan3xp_keys_cursor INTO
    keys ;
WHILE @@FETCH_STATUS = 0 begin
    . . .
END;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field
count	A number defining the actual number of keys returned for this name or address
keys	An array of keys returned for this name or address

## Return Code

negative for error, 0 for success

## ssan3xp\_match

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

### Prototype

```
EXEC master.dbo.ssan3xp_match (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @system , - String in
    @population , - String in
                                @controls , - String in
                                @rsp OUTPUT, - String out
                                @ssaMsg OUTPUT, - String out
                                @search , - EncodedString in
                                @file , - EncodedString in
                                @score OUTPUT, - String out
                                @decision OUTPUT - String out
);
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section
file	Refer to the Common Parameters section
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

### Return Code

negative for error, 0 for success

## ssan3xp\_match\_encoded\_text

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

## Prototype

```
EXEC master.dbo.ssan3xp_match_encoded_text (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @system , - String in
    @population , - String in
    @controls , - String in
    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @search , - EncodedString in
    @file , - EncodedString in
    @score OUTPUT, - String out
    @decision OUTPUT - String out
) ;
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section
file	Refer to the Common Parameters section
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

## Return Code

negative for error, 0 for success

## ssan3xp\_match\_encoded\_unicode

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

## Prototype

```
EXEC master.dbo.ssan3xp_match_encoded_unicode (
    @rc OUTPUT,
    @sockh ,
    @session OUTPUT, - Long io
    @system , - String in
    @population , - String in
    @controls , - String in
```

```

    @rsp OUTPUT, - String out
    @ssaMsg OUTPUT, - String out
    @search , - EncodedString in
    @file , - EncodedString in
    @score OUTPUT, - String out
    @decision OUTPUT - String out
) ;

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section
file	Refer to the Common Parameters section
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

## Return Code

negative for error, 0 for success

## ssan3xp\_open

An optional function, but its use will improve performance. It opens and initiates an SSA-NAME3 session in preparation.

## Prototype

```

using ssa;

$ssa->open (
    $system , # String in
    $population , # String in
    $controls # String in
) ;

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section

controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## ssan3xp\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

## Prototype

```

CREATE TABLE #ssan3xp_ranges_layout (
    stab VARCHAR(256))
INSERT #ssan3xp_ranges_layout
    EXEC master.dbo.ssan3xp_ranges_layout
        @rc OUTPUT,
        @sockh ,
        @session OUTPUT, - Long io
        @system , - String in
        @population , - String in
        @controls , - String in
        @rsp OUTPUT, - String out
        @ssaMsg OUTPUT, - String out
        @data , - Block in
        @encoding , - String in
        @count OUTPUT, - Long out
        @stab OUTPUT, - BlockArray out
        @stab_num ,
        @stab_size ;

if @rc < 0 begin
    goto ret ;
end ;
DECLARE ssan3xp_ranges_cursor CURSOR SCROLL FOR
SELECT
    stab
FROM #ssan3xp_ranges
OPEN ssan3xp_ranges_cursor ;
FETCH FIRST FROM ssan3xp_ranges_cursor INTO
    stab ;
WHILE @@FETCH_STATUS = 0 begin
    . . .
END;

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section



ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs returned for this name or address

### Return Code

negative for error, 0 for success

## Calling from Perl

### addr\_get\_cass\_field

Use this function to retrieve cass specific address fields.

#### Prototype

```
using ssa ;

my $field_value = $ssa->addr_get_cass_field (
    $suggest_idx , # Long in
    $field_idx   # Long in
) ;
```

#### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a cass field
field_idx	Specifies a cass field within the nth suggestion
field_value	The cass field value

### Return Code

negative for error, 0 for success

### addr\_get\_cass\_field\_cnt

Use this function to determine the max number of cass address fields.

## Prototype

```
using ssa ;  
my $count = $ssa->addr_get_cass_field_cnt ( ) ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of cass address fields

## Return Code

negative for error, 0 for success

## addr\_get\_cass\_field\_info

Use this function to retrieve information about a suggestion.

## Prototype

```
using ssa ;  
my @field_length = $ssa->addr_get_cass_field_info (   
    $suggest_idx # Long in  
    ) ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each cass address field

## Return Code

negative for error, 0 for success

## addr\_get\_del\_lines

(deprecated) Use this function to retrieve delivery address line information This function is deprecated. Use **addr\_get\_del\_lines\_ext** instead.

## Prototype

```
using ssa ;
```

```

my ($del_line1 , $del_line2 , $del_line3 , $del_line4 , $del_line5 , $del_line 6 ) =
  $ssa ->addr_get_de
  $suggest_idx # Long in
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string
del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string
del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

## Return Code

negative for error, 0 for success

## addr\_get\_del\_lines\_ext

Use this function to retrieve delivery address line information.

## Prototype

```

using ssa ;

my ($del_line1 , $del_line2 , $del_line3 , $del_line4 , $del_line5 , $del_line6) = $ssa-
>addr_get_de
    $suggest_idx , # Long in
    $del_case     # Long in
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_case	Specifies delivery address line case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case

del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string
del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string
del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

### Return Code

negative for error, 0 for success

## addr\_get\_field

Use this function to retrieve validated address fields.

### Prototype

```
using ssa ;

my ($field_value , $field_val_status , $field_val_mods ) = $ssa->addr_get_field (
    $suggest_idx , # Long in
    $field_idx    # Long in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a field
field_idx	Specifies a field within the nth suggestion
field_value	The field value
field_val_status	Specifies how this field matched the validation data
field_val_mods	Specifies how this field was modified by validation data

### Return Code

negative for error, 0 for success

## addr\_get\_field\_count

Use this function to determine the max number of address fields.

## Prototype

```
using ssa;

my $count = $ssa->addr_get_field_count ( ) ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of address fields

## Return Code

negative for error, 0 for success

## addr\_get\_field\_ext

Use this function to retrieve all getter fields.

## Prototype

```
using ssa;

my $field_value = $ssa->addr_get_field_ext (
    $suggest_idx ,      # Long in
    $field_operation , # Long in
    $field_name ,      # String in
    $field_item_line , # Long in
    $field_type       # String in
) ;
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the section SSA-NAME3 Error Messages
suggest_idx	Specifies the suggestion from which to get fields
field_operation	Field operation Option 0 for AddressElements Option 1 for AddressLines Option 2 for AddressComplete Option 3 for EnrichmentData Option 4 for ResultDataParameter Option 5 for EnrichmentDataStatus
field_name	Refer AD Result.dtd for field names
field_item_line	Represent field line number or field item number
field_type	Refer AD Result.dtd for field attribute Type
field_value	Cleansed field output

## Return Code

negative for error, 0 for success

## addr\_get\_field\_idx

Use this function to retrieve individual address fields.

### Prototype

```
using ssa ;
my $field_value = $ssa->addr_get_field_idx (
    $suggest_idx , # Long in
    $field_idx    # Long in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a field. On successful parse, use 0 for ASM for AD version 4, 1 for ASM for AD version 5
field_idx	Specifies a field within the nth suggestion
field_value	The field value

## Return Code

negative for error, 0 for success

## addr\_get\_field\_info\_ext

Use this function to retrieve information about a suggestion.

### Prototype

```
using ssa ;
my ($field_length , $addr_label_encoded , $addr_label_charset , $score ) = $ssa-
>addr_get_field_info_ext (
    $suggest_idx # Long in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information

field_length	An array containing the length of each address field
addr_label_encoded	The returned label
addr_label_charset	The character set used in the address label
score	The returned label's score

### Return Code

negative for error, 0 for success

## addr\_get\_field\_len

Use this function to determine the max field length.

### Prototype

```
using ssa ;
my $max_len = $ssa->addr_get_field_len ( ) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max address field length in bytes

### Return Code

negative for error, 0 for success

## addr\_get\_line\_len

### Prototype

```
using ssa ;
my $max_len = $ssa->addr_get_line_len ( ) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section

ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max line length in bytes

### Return Code

negative for error, 0 for success

## addr\_get\_option

Use this function to set optional parameters.

### Prototype

```
using ssa ;

my $value = $ssa->addr_get_option (
    $param      # String in
) ;
```

### Parameters

sessrspion	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to get
value	Returns the value for the option

### Return Code

negative for error, 0 for success

## addr\_init

Use this function to initialize the Address Standardization interface.

### Prototype

```
using ssa;

$ssa->addr_init (
    $max_memory # Long in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section



ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_memory	This field specifies the maximum amount of memory (MB) to be used by the Address Standarization engine

### Return Code

negative for error, 0 for success

## addr\_parse

Use this function to parse an address.

### Prototype

```
using ssa ;
my @field_length = $ssa->addr_parse ( ) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_length	An array containing the length of each parsed field

### Return Code

negative for error, 0 for success

## addr\_preload\_country

Use this function to preload country database.

### Prototype

```
using ssa ;
$ssa->addr_preload_country (
    $preload_type , # String in
    $preload_country , # String in
    $val_mode # String in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

preload_type	Type of preload to perform
preload_country	country database to be preloaded
val_mode	This field specified validation mode

### Return Code

negative for error, 0 for success

## addr\_set\_attrb

Use this function to specify the character set of the data and a default country.

### Prototype

```
using ssa;

$ssa->addr_set_attrb (
    $char_set ,      # String in
    $default_country # String in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
char_set	The name of the character set used to encode the input and output
default_country	The default country used for validation when parsing cannot detect a country name

### Return Code

negative for error, 0 for success

## addr\_set\_del\_lines

Use this function to set delivery address line information.

### Prototype

```
using ssa;

$ssa->addr_set_del_lines (
    $del_line1 , # Block in
    $del_line1_size ,
    $del_line2 , # Block in
    $del_line2_size ,
    $del_line3 , # Block in
    $del_line3_size ,
    $del_line4 , # Block in
    $del_line4_size ,
```

```

    $del_line5 ,    # Block in
    $del_line5_size ,
    $del_line6 ,    # Block in
    $del_line6_size
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
del_line1	Delivery address line 1 input string
del_line2	Delivery address line 2 input string
del_line3	Delivery address line 3 input string
del_line4	Delivery address line 4 input string
del_line5	Delivery address line 5 input string
del_line6	Delivery address line 6 input string

## Return Code

negative for error, 0 for success

# addr\_set\_field\_case

Use this function to set individual input fields case option.

## Prototype

```

using ssa;

$ssa->addr_set_field_case (
                                $field_idx , # Long in
                                $field_case # Long in
) ;

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_case	Specifies output field case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case

## Return Code

negative for error, 0 for success

## addr\_set\_field\_idx

Use this function to set individual input fields by idx.

### Prototype

```
using ssa;  
  
$ssa->addr_set_field_idx (  
    $field_idx , # Long in  
    $field_value , # Block in  
    $field_value_size  
);
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_value	Specifies a value for the nth field

### Return Code

negative for error, 0 for success

## addr\_set\_field\_name

Use this function to set individual input fields by name.

### Prototype

```
using ssa;  
  
$ssa->addr_set_field_name (  
    $field_name , # String in  
    $field_value , # Block in  
    $field_value_size  
);
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_name	Specifies the name of the field to set
field_value	Specifies a value for the field

### Return Code

negative for error, 0 for success

## addr\_set\_lines

Use this function to provide an address to parse or validate.

### Prototype

```
using ssa;

$ssa->addr_set_lines (
    $line_1 ,      # Block in
    $line_1_size ,
    $line_2 ,      # Block in
    $line_2_size ,
    $line_3 ,      # Block in
    $line_3_size ,
    $line_4 ,      # Block in
    $line_4_size ,
    $line_5 ,      # Block in
    $line_5_size ,
    $line_6 ,      # Block in
    $line_6_size ,
    $line_7 ,      # Block in
    $line_7_size ,
    $line_8 ,      # Block in
    $line_8_size ,
    $line_9 ,      # Block in
    $line_9_size ,
    $line_10 ,     # Block in
    $line_10_size
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
line_1	The first line of the address
line_2	The second line of the address
line_3	The third line of the address
line_4	The fourth line of the address
line_5	The fifth line of the address
line_6	The sixth line of the address
line_7	The seventh line of the address
line_8	The eighth line of the address
line_9	The ninth line of the address
line_10	The tenth line of the address

### Return Code

negative for error, 0 for success

## addr\_set\_option

Use this function to set optional parameters.

### Prototype

```
using ssa;

$ssa->addr_set_option (
    $param, # String in
    $value # String in
) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to set
value	This field specifies a value for the option

### Return Code

negative for error, 0 for success

## addr\_validate

Use this function to validate an address.

### Prototype

```
using ssa;
my ($status $n_suggest ) = $ssa->addr_validate ( ) ;
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
status	The status returned by the validation process
n_suggest	The number of suggestions generated by validation

### Return Code

negative for error, 0 for success

## close

Closes the SSA session and deallocates memory. Similarly to **ssa\_open**, it is optional.

## Prototype

```
using ssa;
$ssa->close (
    $controls # String in
);
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## convert\_keys

Used to convert 8 byte keys (from server) to 5 byte keys (for client).

## Prototype

```
using ssa;
my @keys5 = $ssa->convert_keys (
    \@keys8 , # StringArray in
    $keys8_num
);
```

## Parameters

keys8	Block of 8 byte keys to be converted
keys5	Block of 5 byte keys (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## convert\_ranges

Used to convert 8 byte ranges (from server) to 5 byte ranges (for client).

## Prototype

```
using ssa;
my @ranges5 = $ssa->convert_ranges (
    \@ranges8 ,      # StringArray in
    $ranges8_num
);
```

## Parameters

ranges8	Block of 8 byte ranges to be converted
ranges5	Block of 5 byte ranges (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

# disconnect

Releases resources allocated to a socket.

## Prototype

```
using ssa;
$ssa->disconnect ( ) ;
```

## Parameters

none

## Return Code

negative for error, 0 for success

# errors\_get\_all

Get the Server side error messages from the last API function that failed. This function should be called repeatedly until it returns 1, meaning all messages have been retrieved.

**Note:** If a communication (socket) error occurred, this function will also fail. Refer to the *OPERATIONS MANUAL*, *Error Log* section for information on interpreting the Error Log.

## Prototype

```
using ssa;
my $msg = $ssa->errors_get_all ( ) ;
```

## Parameters

msg	This is an error message
-----	--------------------------



## Return Code

negative for error, 0 for success

## get\_keys\_encoded

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```
using ssa;

my @keys = $ssa->get_keys_encoded (
    $controls ,          # String in
    $data ,              #
    EncodedString in
    $dataEncType
) ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

## Return Code

negative for error, 0 for success

## get\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```
using ssa;

my @stab = $ssa->get_ranges (
    $controls ,          # String in
    $data                # String in
) ;t
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

## Return Code

negative for error, 0 for success

## info

Returns Information regarding the selected `System` and `Population`.

### Prototype

```
using ssa;

my @info = $ssa->info (
    $controls      # String in
);
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	A number defining the number of rows in the Items Array
info	An array of 1024 byte rows, containing the data returned for the specific <code>ssan3_info</code> call

## Return Code

negative for error, 0 for success

## keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```
using ssa;

my @keys = $ssa->keys (
    $controls ,      # String in
    $data ,          # Block in
    $data_size ,
    $encoding        # String in
) ;
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field
count	A number defining the actual number of keys returned for this name or address
keys	An array of keys returned for this name or address

## Return Code

negative for error, 0 for success

## match

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

### Prototype

```
using ssa;

my ($score , $decision ) = $ssa->match (
                                $controls ,      # String in
```

```

        $search ,           # String in
        $file              # String in
    ) ;

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section
file	Refer to the Common Parameters section
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

## Return Code

negative for error, 0 for success

## match\_encoded

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

## Prototype

```

using ssa;

my ($score , $decision ) = $ssa->match_encoded (
    $controls ,           # String in
    $search ,           #
    EncodedString in
    $searchEncType ,
    $file ,             #
    EncodedString in
    $fileEncType
) ;

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section

population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section
file	Refer to the Common Parameters section
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

### Return Code

negative for error, 0 for success

## ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```
using ssa;

my @stab = $ssa->ranges (
    $controls ,      # String in
    $data ,          # Block in
    $data_size ,    # String in
    $encoding        # String in
);
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field

count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs returned for this name or address

### Return Code

negative for error, 0 for success

## Calling from Visual Basic 6

VB programs should call the SSA-NAME3 DLL (ssan3ca.dll). This is achieved by adding the supplied module %SSATOP%\h\ssan3v6.bas to the VB project concerned.

### addr\_get\_cass\_field

Use this function to retrieve cass specific address fields.

#### Prototype

```
Public Function ssan3_addr_get_cass_field ( _
    ByVal session As Integer , _ ' Long io
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _
    ByRef suggest_idx As Integer , _ ' Long in
    ByRef field_idx As Integer , _ ' Long in
    ByVal field_value ( ) As Byte , _ ' Block out
    ByRef field_value_size As Integer _
) As Integer
```

#### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a cass field
field_idx	Specifies a cass field within the nth suggestion
field_value	The cass field value

### Return Code

negative for error, 0 for success

### ssan3\_addr\_get\_cass\_field\_cnt

Use this function to determine the max number of cass address fields.

## Prototype

```
Public Function ssan3_addr_get_cass_field_cnt ( _  
    ByVal session As Integer , _ ' Long io  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer , _  
    ByVal count As Integer _ ' Long out  
) As Integer
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of cass address fields

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_cass\_field\_info

Use this function to retrieve information about a suggestion.

## Prototype

```
Public Function ssan3_addr_get_cass_field_info ( _  
    ByVal session As Integer , _ ' Long io  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer , _  
    ByRef suggest_idx As Integer , _ ' Long in  
    ByVal field_length ( ) As Integer , _ ' LongArray out  
    ByRef field_length_num As Integer _  
) As Integer)
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each cass address field

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_del\_lines

(deprecated) Use this function to retrieve delivery address line information. This function is deprecated, use **addr\_get\_del\_lines\_ext** instead.

### Prototype

```
Public Function ssan3_addr_get_del_lines ( _  
    ByVal session As Integer , _ ' Long io  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer , _  
    ByRef suggest_idx As Integer , _ ' Long in  
    ByVal del_line1 ( ) As Byte , _ ' Block out  
    ByRef del_line1_size As Integer , _  
    ByVal del_line2 ( ) As Byte , _ ' Block out  
    ByRef del_line2_size As Integer , _  
    ByVal del_line3 ( ) As Byte , _ ' Block out  
    ByRef del_line3_size As Integer , _  
    ByVal del_line4 ( ) As Byte , _ ' Block out  
    ByRef del_line4_size As Integer , _  
    ByVal del_line5 ( ) As Byte , _ ' Block out  
    ByRef del_line5_size As Integer , _  
    ByVal del_line6 ( ) As Byte , _ ' Block out  
    ByRef del_line6_size As Integer _  
    ) As Integer
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string
del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string
del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_del\_lines\_ext

Use this function to retrieve delivery address line information.

### Prototype

```
Public Function ssan3_addr_get_del_lines_ext ( _
```



```

ByVal session As Integer , _ ' Long io
ByVal rsp As String , _ ' String out
ByRef rsp_size As Integer , _ ' String out
ByVal ssaMsg As String , _ ' String out
ByRef ssaMsg_size As Integer , _ ' Long in
ByRef suggest_idx As Integer , _ ' Long in
ByRef del_case As Integer , _ ' Long in
ByVal del_line1 ( ) As Byte , _ ' Block out
ByRef del_line1_size As Integer , _ ' Block out
ByVal del_line2 ( ) As Byte , _ ' Block out
ByRef del_line2_size As Integer , _ ' Block out
ByVal del_line3 ( ) As Byte , _ ' Block out
ByRef del_line3_size As Integer , _ ' Block out
ByVal del_line4 ( ) As Byte , _ ' Block out
ByRef del_line4_size As Integer , _ ' Block out
ByVal del_line5 ( ) As Byte , _ ' Block out
ByRef del_line5_size As Integer , _ ' Block out
ByVal del_line6 ( ) As Byte , _ ' Block out
ByRef del_line6_size As Integer , _
) As Integer

```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_case	Specifies delivery address line case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case
del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string
del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string
del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field

Use this function to retrieve validated address fields.

### Prototype

```

Public Function ssan3_addr_get_field (
ByVal session As Integer , _ ' Long io
ByVal rsp As String , _ ' String out
ByRef rsp_size As Integer , _

```

```

        ByVal ssaMsg As String , _ ' String out
        ByRef ssaMsg_size As Integer , _ ' Long in
        ByRef suggest_idx As Integer , _ ' Long in
        ByRef field_idx As Integer , _ ' Long in
        ByVal field_value ( ) As Byte , _ ' Block out
        ByRef field_value_size As Integer , _ ' Long out
        ByVal field_val_status As Integer , _ ' Long out
        ByVal field_val_mods As Integer _ ' Long out
    ) As Integer

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a field
field_idx	Specifies a field within the nth suggestion
field_value	The field value
field_val_status	Specifies how this field matched the validation data
field_val_mods	Specifies how this field was modified by validation data

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_count

Use this function to determine the max number of address fields.

### Prototype

```

Public Function ssan3_addr_get_field_count (
    ByVal session As Integer , _ ' Long io
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _ ' Long in
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _ ' Long in
    ByVal count As Integer _ ' Long out
) As Integer

```

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of address fields

## Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_ext

Use this function to retrieve all getter fields.

### Prototype

```
Public Function ssan3_addr_get_field_ext ( _  
    ByVal session As Integer , _ ' Long i o  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer , _  
    ByRef suggest_idx As Integer , _ ' Long in  
    ByRef field_operation As Integer , _ ' Long in  
    ByRef field_name As String , _ ' String in  
    ByRef field_item_line As Integer , _ ' Long in  
    ByRef field_type As String , _ ' String in  
    ByVal field_value ( ) As Byte , _ ' Block out  
    ByRef field_value_size As Integer _  
    ) As Integer
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get fields
field_operation	Field operation Option 0 for AddressElements Option 1 for AddressLines Option 2 for AddressComplete Option 3 for EnrichmentData Option 4 for ResultDataParameter Option 5 for EnrichmentDataStatus
field_name	Refer AD Result.dtd for field names
field_item_line	Represent field line number or field item number
field_type	Refer AD Result.dtd for field attribute Type
field_value	Cleansed field output

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_idx

Use this function to retrieve individual address fields.

### Prototype

```
Public Function ssan3_addr_get_field_idx ( _  
    ByVal session As Integer , _ ' Long i o  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer , _  
    ByRef suggest_idx As Integer , _ ' Long in  
    ByRef field_idx As Integer , _ ' Long in
```

```

ByVal field_value ( ) As Byte , _ ' Block out
ByRef field_value_size As Integer _
) As Integer

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a field. On successful parse, use 0 for ASM for AD version 4, 1 for ASM for AD version 5
field_idx	Specifies a field within the nth suggestion
field_value	The field value

## Return Code

negative for error, 0 for success

# ssan3\_addr\_get\_field\_info\_ext

Use this function to retrieve information about a suggestion.

## Prototype

```

Public Function ssan3_addr_get_field_info_ext ( _
ByVal session As Integer , _ ' Long io
ByVal rsp As String , _ ' String out
ByRef rsp_size As Integer , _
ByVal ssaMsg As String , _ ' String out
ByRef ssaMsg_size As Integer , _
ByRef suggest_idx As Integer , _ ' Long in
ByVal field_length ( ) As Integer , _ ' LongArray out
ByRef field_length_num As Integer , _
ByVal addr_label_encoded ( ) As Byte , _ ' Block out
ByRef addr_label_encoded_size As Integer , _
ByVal addr_label_charset As String , _ ' String out
ByRef addr_label_charset_size As Integer , _
ByVal score As Integer _ ' Long out
) As Integer

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each address field
addr_label_encoded	The returned label

addr_label_charset	The character set used in the address label
score	The returned label's score

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_field\_len

Use this function to determine the max field length.

### Prototype

```
Public Function ssan3_addr_get_field_len ( _
    ByVal session As Integer , _ ' Long io
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _
    ByVal max_len As Integer _ ' Long out
) As Integer
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max address field length in bytes

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_line\_len

### Prototype

```
Public Function ssan3_addr_get_line_len ( _
    ByVal session As Integer , _ ' Long io
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _
    ByVal max_len As Integer _ ' Long out
) As Integer
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section

ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max line length in bytes

### Return Code

negative for error, 0 for success

## ssan3\_addr\_get\_option

Use this function to set optional parameters.

### Prototype

```
Public Function ssan3_addr_get_option ( _
    ByVal session As Integer , _ ' Long io
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _
    ByRef param As String , _ ' String in
    ByVal value As String , _ ' String out
    ByRef value_size As Integer _
) As Integer
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to get
value	Returns the value for the option

### Return Code

negative for error, 0 for success

## ssan3\_addr\_init

Use this function to initialize the Address Standardization interface.

### Prototype

```
Public Function ssan3_addr_init ( _
    ByVal session As Integer , _ ' Long io
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _
    ByRef max_memory As Integer _ ' Long in
) As Integer
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_memory	This field specifies the maximum amount of memory (MB) to be used by the Address Standardization engine

## Return Code

negative for error, 0 for success

## ssan3\_addr\_parse

Use this function to parse an address.

### Prototype

```
Public Function ssan3_addr_parse ( _  
    ByVal session As Integer , _ ' Long io  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer , _  
    ByVal field_length ( ) As Integer , _ ' LongArray out  
    ByRef field_length_num As Integer _  
    ) As Integer
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_length	An array containing the length of each parsed field

## Return Code

negative for error, 0 for success

## ssan3\_addr\_preload\_country

Use this function to preload country database.

### Prototype

```
Public Function ssan3_addr_preload_country ( _  
    ByVal session As Integer , _ ' Long io  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer , _  
    ByRef preload_type As String , _ ' String in
```

```

ByRef preload_country As String , _ ' String in
ByRef val_mode As String _ ' String in
) As Integer

```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
preload_type	Type of preload to perform
preload_country	Country database to be preloaded
val_mode	This field specified validation mode

### Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_attrib

Use this function to specify the character set of the data and a default country.

### Prototype

```

Public Function ssan3_addr_set_attrib ( _
ByVal session As Integer , _ ' Long io
ByVal rsp As String , _ ' String out
ByRef rsp_size As Integer , _
ByVal ssaMsg As String , _ ' String out
ByRef ssaMsg_size As Integer , _
ByRef char_set As String , _ ' String in
ByRef default_country As String _ ' String in
) As Integer

```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
char_set	The name of the character set used to encode the input and output
default_country	The default country used for validation when parsing cannot detect a country name

### Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_del\_lines

Use this function to set delivery address line information.



## Prototype

```
Public Function ssan3_addr_set_del_lines ( _
    ByVal session As Integer , _ ' Long io
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _
    ByRef del_line1 ( ) As Byte , _ ' Block in
    ByRef del_line2 ( ) As Byte , _ ' Block in
    ByRef del_line3 ( ) As Byte , _ ' Block in
    ByRef del_line4 ( ) As Byte , _ ' Block in
    ByRef del_line5 ( ) As Byte , _ ' Block in
    ByRef del_line6 ( ) As Byte _ ' Block in
) As Integer
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
del_line1	Delivery address line 1 input string
del_line2	Delivery address line 2 input string
del_line3	Delivery address line 3 input string
del_line4	Delivery address line 4 input string
del_line5	Delivery address line 5 input string
del_line6	Delivery address line 6 input string

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_field\_case

Use this function to set individual input fields case option.

## Prototype

```
Public Function ssan3_addr_set_field_case ( _
    ByVal session As Integer , _ ' Long io
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _
    ByRef field_idx As Integer , _ ' Long in
    ByRef field_case As Integer _ ' Long in
) As Integer
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_case	Specifies output field case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_field\_idx

Use this function to set individual input fields by idx.

### Prototype

```
Public Function ssan3_addr_set_field_idx ( _  
    ByVal session As Integer , _ ' Long io  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer , _  
    ByRef field_idx As Integer , _ ' Long in  
    ByRef field_value ( ) As Byte _ ' Block in  
    ) As Integer
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_value	Specifies a value for the nth field

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_field\_name

Use this function to set individual input fields by name.

### Prototype

```
Public Function ssan3_addr_set_field_name ( _  
    ByVal session As Integer , _ ' Long io
```

```

ByVal rsp As String , _ ' String out
ByRef rsp_size As Integer , _ ' Integer out
ByVal ssaMsg As String , _ ' String out
ByRef ssaMsg_size As Integer , _ ' Integer out
ByRef field_name As String , _ ' String in
ByRef field_value ( ) As Byte _ ' Block in
) As Integer

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_name	Specifies the name of the field to set
field_value	Specifies a value for the field

## Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_lines

Use this function to provide an address to parse or validate.

## Prototype

```

Public Function ssan3_addr_set_lines (
ByVal session As Integer , _ ' Long io
ByVal rsp As String , _ ' String out
ByRef rsp_size As Integer , _ ' Integer out
ByVal ssaMsg As String , _ ' String out
ByRef ssaMsg_size As Integer , _ ' Integer out
ByRef line_1 ( ) As Byte , _ ' Block in
ByRef line_2 ( ) As Byte , _ ' Block in
ByRef line_3 ( ) As Byte , _ ' Block in
ByRef line_4 ( ) As Byte , _ ' Block in
ByRef line_5 ( ) As Byte , _ ' Block in
ByRef line_6 ( ) As Byte , _ ' Block in
ByRef line_7 ( ) As Byte , _ ' Block in
ByRef line_8 ( ) As Byte , _ ' Block in
ByRef line_9 ( ) As Byte , _ ' Block in
ByRef line_10 ( ) As Byte _ ' Block in
) As Integer

```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
line_1	The first line of the address
line_2	The second line of the address

line_3	The third line of the address
line_4	The fourth line of the address
line_5	The fifth line of the address
line_6	The sixth line of the address
line_7	The seventh line of the address
line_8	The eighth line of the address
line_9	The ninth line of the address
line_10	The tenth line of the address

### Return Code

negative for error, 0 for success

## ssan3\_addr\_set\_option

Use this function to set optional parameters.

### Prototype

```
Public Function ssan3_addr_set_option ( _
    ByVal session As Integer , _ ' Long io
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _
    ByRef param As String , _ ' String in
    ByRef value As String _ ' String in
) As Integer
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to set
value	This field specifies a value for the option

### Return Code

negative for error, 0 for success

## ssan3\_addr\_validate

Use this function to validate an address.

## Prototype

```
Public Function ssan3_addr_validate ( _  
    ByVal session As Integer , _ ' Long io  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer , _  
    ByVal status As Integer , _ ' Long out  
    ByVal n_suggest As Integer _ ' Long out  
    ) As Integer
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
status	The status returned by the validation process
n_suggest	The number of suggestions generated by validation

## Return Code

negative for error, 0 for success

## ssan3\_close

Closes the SSA session and deallocates memory. Similarly to **ssa\_open**, it is optional.

## Prototype

```
Public Function ssan3_close ( _  
    ByVal session As Integer , _ ' Long io  
    ByRef system As String , _ ' String in  
    ByRef population As String , _ ' String in  
    ByRef controls As String , _ ' String in  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer _  
    ) As Integer
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## ssan3\_connect

Initiates a socket.

### Prototype

```
Public Function ssan3_connect ( _  
    ByRef host As String , _ ' String in  
    ByRef port As Integer , _ ' Long in  
    ByVal sockh As Integer _ ' Long out  
    ) As Integer
```

### Parameters

host	This is the host to connect to
port	This is the port to connect to
sockh	This is a socket handle

## Return Code

negative for error, 0 for success

## ssan3\_convert\_keys

Used to convert 8 byte keys (from server) to 5 byte keys (for client).

### Prototype

```
Public Function ssan3_convert_keys ( _  
    ByRef keys8 ( ) As String , _ ' StringArray in  
    ByVal keys5 ( ) ( ) As Byte , _ ' BlockArray out  
    ByRef keys5_num As Integer , _  
    ByRef keys5_size As Integer , _  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer _  
    ) As Integer
```

### Parameters

keys8	Block of 8 byte keys to be converted
keys5	Block of 5 byte keys (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## ssan3\_convert\_ranges

Used to convert 8 byte ranges (from server) to 5 byte ranges (for client).

### Prototype

```
Public Function ssan3_convert_ranges ( _  
    ByRef ranges8 ( ) As String , _ ' StringArray in  
    ByVal ranges5 ( ) ( ) As Byte , _ ' BlockArray out  
    ByRef ranges5_num As Integer , _  
    ByRef ranges5_size As Integer , _  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer _  
    ) As Integer
```

### Parameters

ranges8	Block of 8 byte ranges to be converted
ranges5	Block of 5 byte ranges (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## ssan3\_disconnect

Releases resources allocated to a socket.

### Prototype

```
Public Function ssan3_disconnect ( _ ) As Integer
```

### Parameters

none

### Return Code

negative for error, 0 for success

## ssan3\_errors\_get\_all

Get the Server side error messages from the last API function that failed. This function should be called repeatedly until it returns 1, meaning all messages have been retrieved.

**Note:** If a communication (socket) error occurred, this function will also fail. Refer to the *OPERATIONS MANUAL, Error Log* section for information on interpreting the Error Log.

### Prototype

```
Public Function ssan3_errors_get_all ( _  
    ByVal msg As String , _ ' String out  
    ByRef msg_size As Integer _  
    ) As Integer
```

## Parameters

msg	This is an error message
-----	--------------------------

## Return Code

negative for error, 0 for success

# ssan3\_get\_keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

## Prototype

```
Public Function ssan3_get_keys ( _  
    ByVal session As Integer , _ ' Long io  
    ByRef system As String , _ ' String in  
    ByRef population As String , _ ' String in  
    ByRef controls As String , _ ' String in  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer , _  
    ByRef data As String , _ ' String in  
    ByVal count As Integer , _ ' Long out  
    ByVal keys ( ) As String , _ ' StringArray out  
    ByRef keys_size As Integer _  
    ) As Integer
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or address

## Return Code

negative for error, 0 for success



## ssan3\_get\_keys\_encoded

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```
Public Function ssan3_get_keys_encoded ( _  
    ByVal session As Integer , _ ' Long io  
    ByRef system As String , _ ' String in  
    ByRef population As String , _ ' String in  
    ByRef controls As String , _ ' String in  
    ByVal rsp As String , _ ' String out  
    ByRef rsp_size As Integer , _  
    ByVal ssaMsg As String , _ ' String out  
    ByRef ssaMsg_size As Integer , _  
    ByVal data As String , _ ' EncodedString in  
    ByVal dataSize As Integer ,  
    ByVal dataEncType As String ,  
    ByVal count As Integer , _ ' Long out  
    ByVal keys ( ) As String , _ ' StringArray out  
    ByRef keys_size As Integer _  
    ) As Integer
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or address

### Return Code

negative for error, 0 for success

## ssan3\_get\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```
Public Function ssan3_get_ranges ( _  
    ByVal session As Integer , _ ' Long io  
    ByRef system As String , _ ' String in  
    ByRef population As String , _ ' String in
```

```

ByRef controls As String , _ ' String in
ByVal rsp As String , _ ' String out
ByRef rsp_size As Integer , _ ' Integer out
ByVal ssaMsg As String , _ ' String out
ByRef ssaMsg_size As Integer , _ ' Integer out
ByRef data As String , _ ' String in
ByVal count As Integer , _ ' Long out
ByVal stab ( ) As String , _ ' StringArray out
ByRef stab_size As Integer _
) As Integer

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or address

## Return Code

negative for error, 0 for success

## ssan3\_get\_ranges\_encoded

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

## Prototype

```

Public Function ssan3_get_ranges_encoded (
    ByVal session As Integer , _ ' Long io
    ByRef system As String , _ ' String in
    ByRef population As String , _ ' String in
    ByRef controls As String , _ ' String in
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _ ' Integer out
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _ ' Integer out
    ByVal data As String , _ ' EncodedString in
    ByVal dataSize As Integer , _ ' Integer out
    ByVal dataEncType As String , _ ' String out
    ByVal count As Integer , _ ' Long out
    ByVal stab ( ) As String , _ ' StringArray out
    ByRef stab_size As Integer _
) As Integer

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or address

## Return Code

negative for error, 0 for success

## ssan3\_info

Returns Information regarding the selected System and Population.

### Prototype

```
Public Function ssan3_info (
    ByVal session As Integer , _ ' Long io
    ByRef system As String , _ ' String in
    ByRef population As String , _ ' String in
    ByRef controls As String , _ ' String in
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _
    ByVal count As Integer , _ ' Long out
    ByVal info ( ) As String , _ ' StringArray out
    ByRef info_size As Integer _
) As Integer
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section

ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	A number defining the number of rows in the Items Array
info	An array of 1024 byte rows, containing the data returned for the specific <b>ssan3_info</b> call

### Return Code

negative for error, 0 for success

## ssan3\_keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```
Public Function ssan3_keys (
    ByVal session As Integer , _ ' Long io
    ByRef system As String , _ ' String in
    ByRef population As String , _ ' String in
    ByRef controls As String , _ ' String in
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _
    ByRef data ( ) As Byte , _ ' Block in
    ByRef encoding As String , _ ' String in
    ByVal count As Integer , _ ' Long out
    ByVal keys ( ) ( ) As Byte , _ ' BlockArray out
    ByRef keys_num As Integer , _
    ByRef keys_size As Integer _
) As Integer
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field
count	A number defining the actual number of keys returned for this name or address
keys	An array of keys returned for this name or address

### Return Code

negative for error, 0 for success

## ssan3\_match

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

### Prototype

```
Public Function ssan3_match ( _
    ByVal session As Integer , _ ' Long io
    ByRef system As String , _ ' String in
    ByRef population As String , _ ' String in
    ByRef controls As String , _ ' String in
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _
    ByRef search As String , _ ' String in
    ByRef file As String , _ ' String in
    ByVal score As String , _ ' String out
    ByRef score_size As Integer , _
    ByVal decision As String , _ ' String out
    ByRef decision_size As Integer _
) As Integer
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section
file	Refer to the Common Parameters section
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

### Return Code

negative for error, 0 for success

## ssan3\_match\_encoded

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

## Prototype

```
Public Function ssan3_match_encoded ( _
    ByVal session As Integer , _ ' Long io
    ByRef system As String , _ ' String in
    ByRef population As String , _ ' String in
    ByRef controls As String , _ ' String in
    ByVal rsp As String , _ ' String out
    ByRef rsp_size As Integer , _
    ByVal ssaMsg As String , _ ' String out
    ByRef ssaMsg_size As Integer , _
    ByVal search As String , _ ' EncodedString in
    ByVal searchSize As Integer ,
    ByVal searchEncType As String ,
    ByVal file As String , _ ' EncodedString in
    ByVal fileSize As Integer ,
    ByVal fileEncType As String ,
    ByVal score As String , _ ' String out
    ByRef score_size As Integer , _
    ByVal decision As String , _ ' String out
    ByRef decision_size As Integer _
) As Integer
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section
file	Refer to the Common Parameters section
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

## Return Code

negative for error, 0 for success

## ssan3\_open

An optional function, but its use will improve performance. It opens and initiates an SSA-NAME3 session in preparation.

## Prototype

```
Public Function ssan3_open ( _
    ByVal session As Integer , _ ' Long io
    ByRef system As String , _ ' String in
```

```

ByRef population As String , _ ' String in
ByRef controls As String , _ ' String in
ByVal rsp As String , _ ' String out
ByRef rsp_size As Integer , _
ByVal ssaMsg As String , _ ' String out
ByRef ssaMsg_size As Integer _
) As Integer

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## ssan3\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

## Prototype

```

Public Function ssan3_ranges ( _
ByVal session As Integer , _ ' Long io
ByRef system As String , _ ' String in
ByRef population As String , _ ' String in
ByRef controls As String , _ ' String in
ByVal rsp As String , _ ' String out
ByRef rsp_size As Integer , _
ByVal ssaMsg As String , _ ' String out
ByRef ssaMsg_size As Integer , _
ByRef data ( ) As Byte , _ ' Block in
ByRef encoding As String , _ ' String in
ByVal count As Integer , _ ' Long out
ByVal stab ( ) ( ) As Byte , _ ' BlockArray out
ByRef stab_num As Integer , _
ByRef stab_size As Integer _
) As Integer

```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section

rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs returned for this name or address

### Return Code

negative for error, 0 for success

## Calling from Visual Basic .NET

VB.NET programs should call the SSA-NAME3 DLL (`ssan3cs.dll`) as shown in the sample program. Further hints are provided below.

To use the `ssan3cs.dll` in a project being created in the Visual Studios.NET IDE, use the following steps as a guideline:

Add a reference to the `ssan3cs.dll`:

- Right-click on the project/solution name in the **Solution Explorer** pane of the IDE. Select **Add Reference...**
- Make sure that the **.NET** tab is the active tab. Use the **Browse** button to find `ssan3cs.dll`
- Highlight the dll in the Explorer window and click **Open**.
- This adds the `ssan3cs.dll` to the Selected Components pane of the **Add Reference** dialog. Click **OK**.

Add the `ssa` namespace to the project/solution:

- Right-Click on the project/solution name in the **Solution Explore** pane of the IDE. Select **Properties**.
- Click Imports under the **Common Properties** folder.
- In the **Namespace:** textbox type `ssa` and click **Add Import**.
- Click **OK** to exit.

Step 2 eliminates the need to have the following in any module:

```
Imports ssa
```

Otherwise, you can skip Step 2 and add the `Imports` statement as above.

### Coding the `ssan3_close` Call outside of the `Finalize()` Method

Calling the **`ssan3_close`** method from a `Finalize()` method is not recommended. This is because it delays the call to **`ssan3_close`** until the runtime runs garbage collection on the released object, which can be anytime after the object has been released by the program.

Instead, it is recommended that the user implement a public cleanup or close method that performs the necessary housekeeping, including a call to `ssan3_close` to release open sessions prior to releasing the object.



## addr\_get\_cass\_field

Use this function to retrieve cass specific address fields.

### Prototype

```
Imports ssa

Public Function addr_get_cass_field ( _
    ByRef suggest_idx As Integer , _ ' Long in
    ByRef field_idx As Integer _ ' Long in
) As Byte ( )
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a cass field
field_idx	Specifies a cass field within the nth suggestion
field_value	The cass field value

### Return Code

negative for error, 0 for success

## addr\_get\_field\_count

Use this function to determine the max number of address fields.

### Prototype

```
Imports ssa

Public Function addr_get_field_count ( ) As Integer
```

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of address fields

### Return Code

negative for error, 0 for success

## addr\_get\_cass\_field\_info

Use this function to retrieve information about a suggestion.

## Prototype

```
Imports ssa

Public Function addr_get_cass_field_info ( _
                                         ByRef suggest_idx As Integer _ ' Long in
) As Integer ( )
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each cass address field

## Return Code

negative for error, 0 for success

## addr\_get\_del\_lines

(deprecated) Use this function to retrieve delivery address line information. This function is deprecated, use **addr\_get\_del\_lines\_ext** instead.

## Prototype

```
Imports ssa

Public Function addr_get_del_lines ( _
                                     ByRef suggest_idx As Integer _ ' Long in
) As addr_get_del_lines_struct
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string
del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string

del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

### Return Code

negative for error, 0 for success

## addr\_get\_del\_lines\_ext

Use this function to retrieve delivery address line information.

### Prototype

```
Imports ssa

Public Function addr_get_del_lines_ext ( _
    ByRef suggest_idx As Integer , _ ' Long in
    ByRef del_case As Integer _ ' Long in
) As addr_get_del_lines_ext_struct
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get delivery address lines
del_case	Specifies delivery address line case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case
del_line1	delivery address line 1 output string
del_line2	delivery address line 2 output string
del_line3	delivery address line 3 output string
del_line4	delivery address line 4 output string
del_line5	delivery address line 5 output string
del_line6	delivery address line 6 output string

### Return Code

negative for error, 0 for success

## addr\_get\_field

Use this function to retrieve validated address fields.

## Prototype

```
Imports ssa
Public Function addr_get_field (
    ByRef suggest_idx As Integer , _ ' Long in
    ByRef field_idx As Integer _ ' Long in
) As addr_get_field_struct
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the nth suggestion from which to get a field
field_idx	Specifies a field within the nth suggestion
field_value	The field value
field_val_status	Specifies how this field matched the validation data
field_val_mods	Specifies how this field was modified by validation data

## Return Code

negative for error, 0 for success

## addr\_get\_field\_count

Use this function to determine the max number of address fields.

## Prototype

```
Imports ssa

Public Function addr_get_field_count ( ) As Integer
```

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	Returns the max number of address fields

## Return Code

negative for error, 0 for success

## addr\_get\_field\_ext

Use this function to retrieve all getter fields.

## Prototype

```
Imports ssa

Public Function addr_get_field_ext ( _
    ByRef suggest_idx As Integer , _ ' Long in
    ByRef field_operation As Integer , _ ' Long in
    ByRef field_name As String , _ ' String in
    ByRef field_item_line As Integer , _ ' Long in
    ByRef field_type As String _ ' String in
) As Byte ( )
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to get fields
field_operation	Field operation Option 0 for AddressElements Option 1 for AddressLines Option 2 for AddressComplete Option 3 for EnrichmentData Option 4 for ResultDataParameter Option 5 for EnrichmentDataStatus
field_name	Refer AD Result.dtd for field names
field_item_line	Represent field line number or field item number
field_type	Refer AD Result.dtd for field attribute Type
field_value	Cleansed field output

## Return Code

negative for error, 0 for success

## addr\_get\_field\_idx

Use this function to retrieve individual address fields.

## Prototype

```
Imports ssa

Public Function addr_get_field_idx ( _
    ByRef suggest_idx As Integer , _ ' Long in
    ByRef field_idx As Integer _ ' Long in
) As Byte ( )
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

suggest_idx	Specifies the nth suggestion from which to get a field. On successful parse, use 0 for ASM for AD version 4, 1 for ASM for AD version 5
field_idx	Specifies a field within the nth suggestion
field_value	The field value

### Return Code

negative for error, 0 for success

## addr\_get\_field\_info\_ext

Use this function to retrieve information about a suggestion.

### Prototype

```
Imports ssa
Public Function addr_get_field_info_ext ( _
                                     ByRef suggest_idx As Integer _ ' Long in
) As addr_get_field_info_ext_struct
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
suggest_idx	Specifies the suggestion from which to retrieve information
field_length	An array containing the length of each address field
addr_label_encoded	The returned label
addr_label_charset	The character set used in the address label
score	The returned label's score

### Return Code

negative for error, 0 for success

## addr\_get\_field\_len

Use this function to determine the max field length.

### Prototype

```
Imports ssa

Public Function addr_get_field_len ( ) As Integer
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max address field length in bytes

## Return Code

negative for error, 0 for success

# addr\_get\_line\_len

## Prototype

```
Imports ssa  
  
Public Function addr_get_line_len ( ) As Integer
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_len	Returns the max line length in bytes

## Return Code

negative for error, 0 for success

# addr\_get\_option

Use this function to set optional parameters.

## Prototype

```
Imports ssa  
Public Function addr_get_option ( _  
    ByRef param As String _ ' String in  
) As String
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

param	This field specifies the name of the option to get
value	Returns the value for the option

### Return Code

negative for error, 0 for success

## addr\_init

Use this function to initialize the Address Standardization interface.

### Prototype

```
Imports ssa
Public Sub addr_init (
    ByRef max_memory As Integer _ ' Long in
)
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
max_memory	This field specifies the maximum amount of memory (MB) to be used by the Address Standardization engine

### Return Code

negative for error, 0 for success

## addr\_parse

Use this function to parse an address.

### Prototype

```
Imports ssa
Public Function addr_parse ( ) As Integer ( )
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_length	An array containing the length of each parsed field

### Return Code

negative for error, 0 for success



## addr\_preload\_country

Use this function to preload country database.

### Prototype

```
Imports ssa
Public Sub addr_preload_country (
    ByRef preload_type As String , _ ' String in
    ByRef preload_country As String , _ ' String in
    ByRef val_mode As String _ ' String in
)
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
preload_type	Type of preload to perform
preload_country	Country database to be preloaded
val_mode	This field specified validation mode

### Return Code

negative for error, 0 for success

## addr\_set\_attrib

Use this function to specify the character set of the data and a default country.

### Prototype

```
Imports ssa
Public Sub addr_set_attrib (
    ByRef char_set As String , _ ' String in
    ByRef default_country As String _ ' String in
)
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
char_set	The name of the character set used to encode the input and output
default_country	The default country used for validation when parsing cannot detect a country name

### Return Code

negative for error, 0 for success

## addr\_set\_del\_lines

Use this function to set delivery address line information.

### Prototype

```
Imports ssa
Public Sub addr_set_del_lines (
    ByRef del_line1 ( ) As Byte , _ ' Block in
    ByRef del_line2 ( ) As Byte , _ ' Block in
    ByRef del_line3 ( ) As Byte , _ ' Block in
    ByRef del_line4 ( ) As Byte , _ ' Block in
    ByRef del_line5 ( ) As Byte , _ ' Block in
    ByRef del_line6 ( ) As Byte _ ' Block in
)
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
del_line1	Delivery address line 1 input string
del_line2	Delivery address line 2 input string
del_line3	Delivery address line 3 input string
del_line4	Delivery address line 4 input string
del_line5	Delivery address line 5 input string
del_line6	Delivery address line 6 input string

### Return Code

negative for error, 0 for success

## addr\_set\_field\_case

Use this function to set individual input fields case option.

### Prototype

```
Imports ssa
Public Sub addr_set_field_case (
    ByRef field_idx As Integer , _ ' Long in
    ByRef field_case As Integer _ ' Long in
)
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section

ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_case	Specifies output field case option value. The allowed values are 0 = Unchanged, 1 = Upper case, 2 = Lower case and 3 = Mixed case

### Return Code

negative for error, 0 for success

## addr\_set\_field\_idx

Use this function to set individual input fields by idx.

### Prototype

```
Imports ssa
Public Sub addr_set_field_idx ( _
    ByRef field_idx As Integer , _ ' Long in
    ByRef field_value ( ) As Byte _ ' Block in
)
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_idx	Specifies the nth field to set
field_value	Specifies a value for the nth field

### Return Code

negative for error, 0 for success

## addr\_set\_field\_name

Use this function to set individual input fields by name.

### Prototype

```
Imports ssa
Public Sub addr_set_f ield_name ( _
    ByRef field_name As String , _ ' String in
    ByRef field_value ( ) As Byte _ ' Block in
)
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
field_name	Specifies the name of the field to set
field_value	Specifies a value for the field

## Return Code

negative for error, 0 for success

## addr\_set\_lines

Use this function to provide an address to parse or validate.

## Prototype

```
Imports ssa
Public Sub addr_set_lines (
    ByRef line_1 ( ) As Byte , _ ' Block in
    ByRef line_2 ( ) As Byte , _ ' Block in
    ByRef line_3 ( ) As Byte , _ ' Block in
    ByRef line_4 ( ) As Byte , _ ' Block in
    ByRef line_5 ( ) As Byte , _ ' Block in
    ByRef line_6 ( ) As Byte , _ ' Block in
    ByRef line_7 ( ) As Byte , _ ' Block in
    ByRef line_8 ( ) As Byte , _ ' Block in
    ByRef line_9 ( ) As Byte , _ ' Block in
    ByRef line_10 ( ) As Byte _ ' Block in
)
```

## Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
line_1	The first line of the address
line_2	The second line of the address
line_3	The third line of the address
line_4	The fourth line of the address
line_5	The fifth line of the address
line_6	The sixth line of the address

line_7	The seventh line of the address
line_8	The eighth line of the address
line_9	The ninth line of the address
line_10	The tenth line of the address

### Return Code

negative for error, 0 for success

## addr\_set\_option

Use this function to set optional parameters.

### Prototype

```
Imports ssa
Public Sub addr_set_option ( _
    ByRef param As String , _ ' String in
    ByRef value As String _ ' String in
)
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
param	This field specifies the name of the option to set
value	This field specifies a value for the option

### Return Code

negative for error, 0 for success

## addr\_validate

Use this function to validate an address.

### Prototype

```
Imports ssa

Public Function addr_validate ( ) As addr_validate_struct
```

### Parameters

session	Refer to the Common Parameters section
rsp	Refer to the Common Parameters section

ssaMsg	Refer to the SSA-NAME3 Error Messages section
status	The status returned by the validation process
n_suggest	The number of suggestions generated by validation

### Return Code

negative for error, 0 for success

## close

Closes the SSA session and deallocates memory. Similarly to **ssa\_open**, it is optional.

### Prototype

```
Imports ssa
Public Sub close ( _
    ByRef controls As String _ ' String in
)
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## ssan3\_convert\_keys

Used to convert 8 byte keys (from server) to 5 byte keys (for client).

### Prototype

```
Imports ssa
Public Function convert_keys ( _
    ByRef keys8 ( ) As String _ ' StringArray in
) As Byte ( ) ( )
```

## Parameters

keys8	Block of 8 byte keys to be converted
keys5	Block of 5 byte keys (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## ssan3\_convert\_ranges

Used to convert 8 byte ranges (from server) to 5 byte ranges (for client).

### Prototype

```
Imports ssa

Public Function convert_ranges ( _
    ByRef ranges8 ( ) As String _ ' StringArray in
) As Byte ( ) ( )
```

## Parameters

ranges8	Block of 8 byte ranges to be converted
ranges5	Block of 5 byte ranges (output)
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

## Return Code

negative for error, 0 for success

## disconnect

Releases resources allocated to a socket.

### Prototype

```
Imports ssa

Public Sub disconnect ( )
```

## Parameters

none

## Return Code

negative for error, 0 for success

## errors\_get\_all

Get the Server side error messages from the last API function that failed. This function should be called repeatedly until it returns 1, meaning all messages have been retrieved.

**Note:** If a communication (socket) error occurred, this function will also fail. Refer to the *OPERATIONS MANUAL, Error Log* section for information on interpreting the Error Log.

### Prototype

```
Imports ssa

Public Function errors_get_all ( ) As String
```

### Parameters

msg	This is an error message
-----	--------------------------

### Return Code

negative for error, 0 for success

## get\_keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```
Imports ssa

Public Function get_keys ( _
    ByRef controls As String , _ ' String in
    ByRef data As String _ ' String in
) As String ( )
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address



## Return Code

negative for error, 0 for success

## ssan3\_get\_keys\_encoded

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```
Imports ssa

Public Function get_keys_encoded (
    ByRef controls As String , _ ' String in
    ByVal -16s As String ' data , EncodedString
    ByVal As Integer dataSize ,
    ByVal As String dataEncType
) As String ( )
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of keys returned for this name or address
keys	An array of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or address

## Return Code

negative for error, 0 for success

## get\_ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```
Imports ssa

Public Function get_ranges (
    ByRef controls As String , _ ' String in
    ByRef data As String _ ' String in
) As String ( )
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the <code>KEY_SIZE</code> Control, returned for this name or address

## Return Code

negative for error, 0 for success

## ssan3\_get\_ranges\_encoded

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

## Prototype

```
Imports ssa

Public Function get_ranges_encoded (
    ByRef controls As String , _
    ByVal -16s As String ' data , EncodedString
    ByVal As Integer dataSize ,
    ByVal As String dataEncType
) As String ( )
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section

count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs of 8 or 5-byte keys, depending on the setting of the KEY_SIZE Control, returned for this name or address

### Return Code

negative for error, 0 for success

## ssan3\_info

Returns Information regarding the selected System and Population.

### Prototype

```
Imports ssa

Public Function info (
    ByRef controls As String _ ' String in
) As String ( )
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
count	A number defining the number of rows in the Items Array
info	An array of 1024 byte rows, containing the data returned for the specific <b>ssan3_info</b> call

### Return Code

negative for error, 0 for success

## keys

Used to get the SSA-NAME3 Keys for a name or address which the application program will store in the SSA Keys table.

### Prototype

```
Imports ssa

Public Function keys ( _
    ByRef controls As String , _ ' String in
    ByRef data ( ) As Byte , _ ' Block in
    ByRef encoding As String _ ' String in
) As Byte ( ) ( )
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field
count	A number defining the actual number of keys returned for this name or address
keys	An array of keys returned for this name or address

## Return Code

negative for error, 0 for success

## match

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

## Prototype

```
Imports ssa
Public Function match ( _
    ByRef controls As String , _ ' String in
    ByRef search As String , _ ' String in
    ByRef file As String _ ' String in
) As match_struct
```

## Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section

file	Refer to the Common Parameters section
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

### Return Code

negative for error, 0 for success

## ssan3\_match\_encoded

Used to get a score and match decision for two records, a search record and a file record. Once a set of candidate records have been retrieved as a result of selecting data based on the ranges returned in **ssan3\_get\_ranges** call, **ssan3\_match** is called to further qualify the candidate records.

### Prototype

```
Imports ssa
Public Function match_encoded (
    ByRef controls As String , _ ' String in
    ByVal -16s As String ' search , EncodedString
    ByVal As Integer sear chSize ,
    ByVal As String searchEncType ,
    ByVal -16s As String ' file , EncodedString
    ByVal As Integer fileSize ,
    ByVal As String fileEncType
) As match_encoded_struct
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section
search	Refer to the Common Parameters section
file	Refer to the Common Parameters section
score	A 3 byte number between 0-100 indicating a confidence level of the match. The score value depends on the data, Match Purpose and Match Level
decision	'A' meaning 'Accept'. The Score is above the Accept Limit for the specified Purpose and Match Level

### Return Code

negative for error, 0 for success

## open

An optional function, but its use will improve performance. It opens and initiates an SSA-NAME3 session in preparation.

### Prototype

```
Imports ssa
Public Sub open ( _
    ByRef system As String , _ ' String in
    ByRef population As String , _ ' String in
    ByRef controls As String _ ' String in
)
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section
ssaMsg	Refer to the SSA-NAME3 Error Messages section

### Return Code

negative for error, 0 for success

## ranges

Used to get the SSA-NAME3 Key Ranges for a name or address which the application program will use in a Select statement to retrieve records from the SSA Keys table.

### Prototype

```
Imports ssa
Public Function ranges ( _
    ByRef controls As String , _ ' String in
    ByRef data ( ) As Byte , _ ' Block in
    ByRef encoding As String _ ' String in
) As Byte ( ) ( )
```

### Parameters

session	Refer to the Common Parameters section
system	Refer to the Common Parameters section
population	Refer to the Common Parameters section
controls	Refer to the Controls section
rsp	Refer to the Common Parameters section

ssaMsg	Refer to the SSA-NAME3 Error Messages section
data	Refer to the REF Controls section
encoding	The encoding used for the data field
count	A number defining the actual number of key ranges returned for this name or address
stab	An array of pairs returned for this name or address

### Return Code

negative for error, 0 for success

## CHAPTER 5

# Controls

When invoking any of the SSA-NAME3 functions, it is necessary to specify the `Controls` to be used. The format and content of the `Controls` is dependent on the function being invoked.

### Conventions

- Controls take the form of keyword=value.
- Multiple Controls are separated by a single space. There are no spaces within the keyword or the value themselves.
- [ ] around a Control means that Control is optional.
- The Controls are not case sensitive.
- Some Control values (example, field names, search names, purpose names) depend upon the Population Rules being used. For a definitive list of the Controls values available in your Population, use the SSA-NAME3 Developer's Workbench.

## ssan3\_connect Controls

`connect` does not require any Controls.

## ssan3\_disconnect Controls

`disconnect` does not require any Controls.

## ssan3\_open Controls

Use optional Controls of the `ssan3_open` function to specify the environment variable, session timeout value, and maximum number of fields in a record.

### Syntax

```
[SSAPR=}directory name]
[TIMEOUT=}n seconds]
```



```
[MAX_FIELDS=}number of fields]
[DISABLESIGHANDLER=Y|N]
```

## Definitions

SSAPR (Optional)	Specify the directory containing the Population Files. For example, on MSWin32 platforms this may be: c:\InformaticaIR\nm3\pr. Or for Unix platforms it may be / InformaticaIR/nm3/pr. This can also be set as an environment variable. <b>Note:</b> You can set the SSAPR Control as an environment variable in the execution environment of the application or SSA-NAME3 server.
TIMEOUT (Optional)	Specify the number of seconds to keep a session since the last API call before allowing it to be overwritten. The default setting is 0 - i.e. no time-out value. This allows inactive sessions to be made available for new sessions, and reduces the risk of SSA-NAME3 running out of sessions. For more information, see the SSA-NAME3 "Sessions" section.
MAX_FIELDS (Optional)	Specify the number of fields in a record to avoid internal errors because of the predefined field limit of 256 set for records. The valid range is from 64 to 65536.
DISABLESIGHANDLER=Y N (Optional)	Applicable to UNIX platforms. Disables the signal handler for the following signals: The signal s, and the signal <ul style="list-style-type: none"> <li>- SIGILL. Reports illegal instructions.</li> <li>- SIGFPE. Reports floating point errors.</li> </ul> Default is Y.

## ssan3\_close Controls

For **ssan3\_close** optional Controls can be used to force a Population to be unloaded from memory and/or force all open sessions closed. In a multi-user environment, these options should only be used by the application responsible for the distribution of sessions to clients. This application is the only component in a position to know when all sessions are closed, or to block new sessions from being opened if the need for either of these functions exists. Improper use of these Controls in a production environment could adversely impact regular application processing. Call Informatica Corporation Technical Support for more information on using these Controls.

### Syntax

```
[UNLOAD]
[TERM]
```

### Definitions

UNLOAD (Optional)	Specifying UNLOAD on an <b>ssan3_close</b> call will cause the Population specified in the call to be unloaded from memory, but only after all currently open sessions have been closed. After a successful unload, the next call to SSA-NAME3 requiring this Population will trigger a reload from disk.
TERM (Optional)	Specifying TERM on an <b>ssan3_close</b> call will force all currently open sessions in the SSA-NAME3 instance to be closed. This will force a reload of the Population on the next call to SSA-NAME3. Using this Control while sessions are open could result in abnormal termination of the processes using those sessions and should be used with great care.

# ssan3\_get\_keys\_encoded Controls

For `ssan3_get_keys_encoded`, the `Controls` specify the fields in the data that you can use as the key fields for key building, the key level, and the layout of the key field data.

## Syntax

```

FIELD=field name
[KEY_LEVEL=key level]
[KEY_SIZE=5|8]
[NAMEFORMAT=name format]
[UNICODE_ENCODING=Unicode type]
[ENCODING=Unicode type]
[LAYOUT=[field1,]offset1,length1,... ,
[fieldn,]offsetn,lengthn]
[DELIMITER=delimiter]
[MAX_ENTRIES=n]
[MUST_RETURNVALUES=Y/N]
[GEOCODE_FORMAT=0|1]
[NM3KEYWORDS=KEY-TABLE|FMTOVERRIDES|FIRSTWORDRANGEORKEY|EXTRACODEPROBES]
[DEBUG_ENABLE=Y/N]
[DEBUG_FILE=file name]
[NONULLKEY=Y/N]

```

## Definitions

Field	Mandatory/Optional	Description
FIELD	Mandatory	<p>Defines the field for which you want to build keys. With most standard populations, you can use one of the following predefined fields:</p> <ul style="list-style-type: none"> <li>- Person_Name</li> <li>- Organization_Name</li> <li>- Address_Part1</li> <li>- Geocode</li> </ul> <p>You can use only one key field in a function call. If you want to build a key on another field that contains a different type of data, use two separate functions so that you can store the keys in two separate indexes.</p>
KEY_LEVEL	Optional	<p>Indicates the type of key level that you want to build. With most standard populations, you can use one of the following predefined values:</p> <ul style="list-style-type: none"> <li>- Standard</li> <li>- Extended</li> <li>- Limited</li> </ul> <p>The default value is Standard.</p>
KEY_SIZE=[5 8]	Optional	<p>Specifies the length of the SSA-NAME3 keys that you want to return. The default key-size for both the <code>ssan3_get_keys</code> and <code>ssan3_get_ranges</code> functions is 8 bytes.</p> <p>For some applications and database designs, you can use a smaller and more compressed key. In this case, specifying <code>KEY_SIZE</code> to 5 instructs SSA-NAME3 to build the 5-byte binary key instead of the 8-byte character key.</p> <p><b>Note:</b> To use the 5-byte keys, check that your database supports the storage and access of binary keys and that your application environment is capable of passing binary values around unchanged.</p> <p>In IBM DB2 UDB, set the <code>IDENTITY</code> option when you create the database so that the collating sequence of the 8-byte character keys is correct. Alternatively, you can use the 5-byte binary keys.</p>

Field	Mandatory/Optional	Description
NAMEFORMAT L/R	Optional	<p>Defines whether the major word in the name or address is on the left end or the right end. For example, in Western person names, the family name is on the right end of the names.</p> <p>Use the NAMEFORMAT control to override the default name format. For more information about the default name format for a given standard population field and its effects, from the <b>Help</b> menu of SSA-NAME3 Workbench, click <b>Population Documentation</b>.</p>
PREFKEYONLY Y/A	Optional	<p>Returns only the preferred key for the input record. If the input record contains multiple values, then PREFKEYONLY=A returns the preferred key for each value. PREFKEYONLY=Y only returns one key regardless of the number of values that pass for the field.</p>
UNICODE_ENCODING	Optional	<p>Instructs SSA-NAME3 to accept Unicode data input, and specifies the Unicode format of the data that you want to pass.</p>
ENCODING	Optional	<p>Functions similar to the UNICODE_ENCODING keyword.</p>
LAYOUT	Optional	<p>Specifies the offset and length pairs (Scatter / Gather format) of the data to be passed for key building in the Key Field Data parameter. Refer to the section on Function parameters for more information. If this control is omitted then the data is expected to be in Tagged Data Format.</p>
DELIMITER	Optional	<p>Overrides the default delimiter - asterisk (*) if you pass the key field data by using the Tagged Data Format.</p>
MAX_ENTRIES	Optional	<p>Controls the maximum number of keys that SSA-NAME3 returns in the Required Keys parameter. By default, SSA-NAME3 returns a maximum of 1024 keys. The control accepts integer values up to 65535. If a name generates more keys than the number specified by MAX_ENTRIES, a Response Code of 1 is returned with an empty Keys array.</p>
MUST_RETURNVALUES=[Y N]	Optional	<p>Forces SSA-NAME3 to return the keys. Set this field to Y when you get a key stack overflow error.</p> <p>Default is N.</p>
GEOCODE_FORMAT={0 1}	Optional	<p>Indicates the order of the latitude and longitude coordinates that you specify to generate keys.</p> <p>If GEOCODE_FORMAT=1, SSA-NAME3 considers the geographic coordinates in the following order: *Geocode* &lt;Longitude&gt; &lt;Latitude&gt;***</p> <p>If GEOCODE_FORMAT=0, SSA-NAME3 considers the geographic coordinates in the following order: *Geocode* &lt;Latitude&gt; &lt;Longitude&gt;***</p> <p>The default value is 0.</p>

Field	Mandatory/Optional	Description
NM3KEYWORDS	Optional	<p>Indicates the additional tasks that SSA-NAME3 must perform when it builds keys. Use one or more of the following keywords:</p> <ul style="list-style-type: none"> <li>- <b>KEY-TABLE.</b> Builds additional data that explains how SSA-NAME3 constructs each key based on the word stack order. You can retrieve this additional data by executing the <code>ssan3_info_call</code> with <code>results.key-table</code> as the Item Type. The following example displays the key details of James Alan Mackintosh: <pre>0 'KTBL: 0524 KTBL18104 019 008' 1 ' 1 00 00 00 03+01+02+00+00 0' 2 ' PART 021 James Alan Mackintosh' 3 ' NAME 020 MCKINTOSH JAMES ALAN'</pre> <p>The value <code>03+01+02+00+00</code> indicates that the key is built based on the word stack order, <code>MCKINTOSH JAMES ALAN</code>.</p> </li> <li>- <b>FMTOVERRIDES.</b> Specifies whether to disable or override a list of category names and category types. Use the following format: <pre>FMTOVERRIDES=*cX,cX,cX,cX*</pre> <p>The format uses the following values:</p> <ul style="list-style-type: none"> <li>- <b>c.</b> Use one of the following values: <ul style="list-style-type: none"> <li>- <b>N</b> - Disables the specified edit-list category names during key generation process.</li> <li>- <b>T</b> - Disables the specified edit-list category types during key generation process.</li> <li>- <b>X</b> - Overrides the specified edit-list category type with another category type.</li> <li>- <b>Y</b> - Overrides the specified edit-list category name with another category type.</li> </ul> </li> <li>- <b>X.</b> Denotes the category name or the category type in the edit-list that you want to disable or override.</li> </ul> <p>The following sample configuration indicates that SSA-NAME3 must override Noise Words category name with Skip category type:</p> <pre>Controls: 'FIELD=Organization_Name NM3KEYWORDS=FMTOVERRIDES=*,,YNWS,*' Key Field Data: '*Organization_Name*The David Ford***'</pre> <p>For the preceding sample configuration, SSA-NAME3 retains the word <code>THE</code> in the word stack because of the value <code>YNWS</code> in the <code>FMTOVERRIDES</code> keyword.</p> <p><b>Note:</b> The <code>FMTOVERRIDES</code> keyword is also applicable for extended fields.</p> </li> <li>- <b>FIRSTWORDRANGEORKEY.</b> Builds an additional key if the first word in the word stack is a skip word and the <code>NAMEFORMAT</code> keyword is set to <code>L</code>.</li> <li>- <b>EXTRACODEPROBES.</b> Builds an additional probe for each code with five or more characters in the word stack after formatting. The keys contain alphanumeric characters. Use this keyword to support model numbers for products that might not have digits.</li> </ul> <p><b>Note:</b> For more information about other valid keywords that the <code>ssan_get_keys_encoded</code> control supports, see the <code>NAMESET</code> function keywords section in the <i>Informatica SSA-NAME3(EXTN) Service Group Definition and Customization Guide</i>.</p>
DEBUG_ENABLE=[Y N]	Optional	<p>Verifies the input parameters and writes debug information about the parameters to a specified log file. Default is <code>N</code>.</p>

Field	Mandatory/Optional	Description
DEBUG_FILE	Optional	When <code>DEBUG_ENABLE</code> is set to <code>Y</code> , <code>DEBUG_FILE</code> specifies the name of the log file to which you want to write the debug information about the input parameters.
NONULLKEY=[Y N]	Optional	Specifies whether to include null keys in the <code>SSA-NAME3</code> functions. When set to <code>N</code> , <code>SSA-NAME3</code> returns a null key. When set to <code>Y</code> , <code>SSA-NAME3</code> doesn't return a null key and returns 0 for the count of keys.

## ssan3\_convert\_keys Controls

`Convert Keys` does not require any Controls.

## ssan3\_get\_ranges\_encoded Controls

For `ssan3_get_ranges_encoded`, the `Controls` specify the fields in the data that you can use as the key fields to build search ranges, the search level, and the layout of the key field data.

### Syntax

```

FIELD=field name
[SEARCH_LEVEL=search level]
[KEY_SIZE=5|8]
[NAMEFORMAT=name format]
[UNICODE_ENCODING=Unicode type]
[ENCODING=Unicode type]
[SEARCH_LIMIT=n]
[LIMIT_MODE=Y/N]
[LAYOUT=[field1,]offset1,length1,... ,
[fieldn,]offsetn,lengthn]
[DELIMITER=delimiter]
[MAX_ENTRIES=n]
[MUST_RETURNVALUES=Y/N]
[SECPROBE=Y/N]
[BATCHMODE=Y|S|N]
[GEOCODE_FORMAT=0|1]
[GEOCODE_RADIUS=n]
[NM3KEYWORDS=FIRSTMINORPROBE|ORIGWORDRANGE|CONCMINORRANGE|INITRANGE|INITPROBE|DEBUG|
FMTOVERRIDES|EXTRADATEDROP1PART|FIRSTWORDRANGEORKEY|BUILDPROBEDROPFIRSTINIT|
EXTRACODEPROBES]
[DEBUG_ENABLE=Y/N]
[DEBUG_FILE=file name]
[NONULLKEY=Y/N]

```

## Definitions

Field	Description
FIELD	<p>Defines the field on which you want to build search ranges. With most standard populations, you can use one of the following predefined fields:</p> <ul style="list-style-type: none"> <li>- Person_Name</li> <li>- Organization_Name</li> <li>- Address_Part1</li> <li>- Geocode</li> </ul> <p>You can use only one key field in a function call.</p>
SEARCH_LEVEL	<p>Optional. Specifies the type of search level that you want to perform. With most standard populations, you can use one of the following predefined values:</p> <ul style="list-style-type: none"> <li>- Typical</li> <li>- Exhaustive</li> <li>- Extreme</li> <li>- Narrow</li> </ul> <p>The default value is Typical.</p> <p><b>Note:</b> The <code>SEARCH_LEVEL</code> control is not applicable to the Geocode field. If you specify the <code>SEARCH_LEVEL</code> control with a Geocode field, SSA-NAME3 ignores the <code>SEARCH_LEVEL</code> control.</p>
KEY_SIZE=[5 8]	<p>Optional. Specifies the length of the SSA-NAME3 keys that you want to return. Use the same value that you have set for the <code>ssan3_get_keys</code> function. The default key-size for both the <code>ssan3_get_keys</code> and <code>ssan3_get_ranges</code> functions is 8 bytes.</p> <p>For some applications and database designs, you can use a smaller and more compressed key. In this case, specifying the <code>KEY_SIZE</code> to 5 instructs SSA-NAME3 to build the 5-byte binary key instead of the 8-byte character key.</p> <p><b>Note:</b> To use the 5-byte keys, check that your database supports the storage and access of binary keys and that your application environment can pass binary values around unchanged. In IBM DB2 UDB, set the <code>IDENTITY</code> option when you create the database so that the collating sequence of the 8-byte character keys is correct. Alternatively, you can use the 5-byte binary keys.</p>
NAMEFORMAT L/R	<p>Optional. Defines whether the major word in the name or address is on the left end or the right end. For example, in Western person names, the family name is on the right end of the names.</p> <p>Use the <code>NAMEFORMAT</code> control to override the default name format. For more information about the default name format for a given standard population field and its effects, from the <b>Help</b> menu of SSA-NAME3 Workbench, click <b>Population Documentation</b>.</p>
UNICODE_ENCODING	<p>Optional. Instructs SSA-NAME3 to accept Unicode data input, and specifies the Unicode format of the data that you want to pass.</p>
ENCODING	<p>Optional. Functions similar to the <code>UNICODE_ENCODING</code> keyword.</p>
SEARCH_LIMIT	<p>Optional. Specifies the maximum percentage of the file that is allowed to be returned in any one search. This can be helpful in preventing costly searches. The percentage value is a number up to 2 decimal places. When the function is called, the Search Strategy is evaluated before any database I/O is done. If it is calculated that the search would return a greater percentage of the file than allowed, an error message will be returned to the calling program.</p> <p><b>Note:</b> For the <code>Search_Limit</code> to be useful, the Population must include a Scalar Frequency Table. See the Population Override Manager for more details.</p>

Field	Description
LIMIT_MODE=[Y/N]	<p>Removes the ranges that exceed the value set by the <code>SEARCH_LIMIT</code> option. If there are no ranges left to return, <code>SSA-NAME3</code> returns the response 1. Use the <code>LIMIT_MODE=Y</code> option in combination with the <code>SEARCH_LIMIT</code> option.</p> <p>The <code>LIMIT_MODE=Y</code> option changes the default behaviour of the <code>SEARCH_LIMIT</code> option. By default, the <code>SEARCH_LIMIT</code> option applies to all the ranges, and if the limit exceeds, <code>SSA-NAME3</code> does not return any ranges. If you set the <code>LIMIT_MODE=Y</code> option, the limit applies to each range and <code>SSA-NAME3</code> removes or discards the ranges that exceed the search limit.</p> <p>Default is N.</p>
MAX_ENTRIES	<p>Controls the maximum number of keys that <code>SSA-NAME3</code> returns in the <code>Required Keys</code> parameter. By default, <code>SSA-NAME3</code> returns a maximum of 1024 keys. The control accepts integer values up to 65535. If a name generates more keys than the number specified by <code>MAX_ENTRIES</code>, a Response Code of 1 is returned with an empty Keys array.</p>
MUST_RETURNVALUE S=[Y N]	<p>Forces <code>SSA-NAME3</code> to return the keys. Set this field to Y when you get a key stack overflow error.</p> <p>Default is N.</p>
SECPROBE Y/N	<p>Optional. Changes secondary name ranges from a range to a probe and narrows the searches that use secondary names.</p>
BATCHMODE=[Y S N]	<p>Optional. Adjusts secondary name searches and custom set ranges for batch applications. You can convert secondary name searches from ranges to probes and custom set search ranges to probes, and you can remove secondary name searches. Use one of the following values:</p> <p><b>Y</b></p> <p>Converts secondary name searches from ranges to probes and custom set ranges to probes.</p> <p><b>S</b></p> <p>Removes all the secondary name searches and converts custome set ranges to probes.</p> <p><b>N</b></p> <p>Turns off BATCHMODE.</p>
GEOCODE_FORMAT=[0 1]	<p>Optional. Indicates the order of the latitude and longitude coordinates that you specify to build search ranges.</p> <p>If <code>GEOCODE_FORMAT=1</code>, <code>SSA-NAME3</code> considers the geographic coordinates in the following order:</p> <p>*Geocode*&lt;Longitude&gt; &lt;Latitude&gt;***</p> <p>If <code>GEOCODE_FORMAT=0</code>, <code>SSA-NAME3</code> considers the geographic coordinates in the following order:</p> <p>*Geocode*&lt;Latitude&gt; &lt;Longitude&gt;***</p> <p>The default value is 0.</p>

Field	Description
GEOCODE_RADIUS	<p>Optional. Indicates the search radius to build search ranges according to the latitude and longitude coordinates that you specify. The default value is 1000 m, and the default unit is meter.</p> <p>For example:</p> <pre>FIELD=GEOCODE GEOCODE_FORMAT=1 GEOCODE_RADIUS=100</pre> <p>Use the following guidelines when you specify the geocode radius:</p> <ul style="list-style-type: none"> <li>- SSA-NAME3 considers the radius to be in meters even if you specify any other unit.</li> <li>- SSA-NAME3 ignores the decimal values and uses the whole number as the radius. For example, if you specify the radius as 850.8, SSA-NAME3 considers the radius as 850.</li> </ul>



Field	Description
NM3KEYWORDS	<p>Optional. Indicates the additional keys that you want to build based on the keywords you specify.</p> <p>Use one or more of the following keywords:</p> <ul style="list-style-type: none"> <li>- <b>FIRSTMINORPROBE.</b> Builds an additional probe for the first minor word. For example, if the name is <b>PEPE JONES</b>, the <b>FIRSTMINORPROBE</b> keyword builds an additional probe for the first minor word, <b>PEPE</b>.</li> <li>- <b>ORIGWORDRANGE.</b> Builds an additional search range based on the unformatted words after cleaning and stabilization, but without processing the Edit-list definitions.</li> <li>- <b>CONCMINORRANGE.</b> Builds an additional search range after concatenating the first two minors. For example, if the name is <b>JONG KU LEE</b> where <b>LEE</b> is the major word, the <b>CONCMINORRANGE</b> keyword builds a search range for <b>JONGKU LEE</b>.</li> <li>- <b>INITRANGE.</b> Builds an additional search range for the acronym of the name. For example, if the name is <b>THE ATLANTIC AND PACIFIC TEA COMPANY</b>, the <b>INITRANGE</b> keyword builds a range for <b>APT*</b>.</li> <li>- <b>INITPROBE.</b> Builds an additional probe for the acronym of the search name. For example, if the name is <b>THE ATLANTIC AND PACIFIC TEA COMPANY</b>, the <b>INITRANGE</b> keyword builds a range for <b>APT!</b>.</li> <li>- <b>DEBUG.</b> Builds additional data that explains how SSA-NAME3 constructs each key based on the word stack order for troubleshooting purposes.</li> <li>- <b>FMTOVERRIDES.</b> Specifies whether to disable or override a list of category names and category types.</li> </ul> <p>Use the following format:  <b>FMTOVERRIDES=*cX, cX, cX, cX*</b></p> <p>The format uses the following values:</p> <ul style="list-style-type: none"> <li>- <b>c.</b> Use one of the following values: <ul style="list-style-type: none"> <li>- <b>N</b> - Disables the specified edit-list category names during key generation process.</li> <li>- <b>T</b> - Disables the specified edit-list category types during key generation process.</li> <li>- <b>X</b> - Overrides the specified edit-list category type with another category type.</li> <li>- <b>Y</b> - Overrides the specified edit-list category name with another category type.</li> </ul> </li> <li>- <b>X.</b> Denotes the category name or the category type in the edit-list that you want to disable or override.</li> </ul> <p>The following sample configuration indicates that SSA-NAME3 must override <b>Noise Words</b> category name with <b>Skip</b> category type:</p> <pre>Controls: 'FIELD=Organization_Name NM3KEYWORDS=FMTOVERRIDES=*, ,YNWS,*' Key Field Data: '*Organization_Name*The David Ford***'</pre> <p>For the preceding sample configuration, SSA-NAME3 retains the word <b>THE</b> in the word stack because of the value <b>YNWS</b> in the <b>FMTOVERRIDES</b> keyword.</p> <p><b>Note:</b> The <b>FMTOVERRIDES</b> keyword is also applicable for extended fields.</p> <ul style="list-style-type: none"> <li>- <b>EXTRADATEDROP1PART.</b> Builds additional keys for dates after splitting the date into multiple parts.</li> </ul> <p>A date in the <b>yyyy-mm-dd</b> format is split into the following parts:</p> <ul style="list-style-type: none"> <li>- <b>yy-mm</b></li> <li>- <b>yy-dd</b></li> <li>- <b>mm-dd</b></li> </ul> <p>For example, if you want to find the dates <b>2000-10-11</b>, <b>2000-10-13</b>, and <b>2002-10-11</b>, the <b>EXTRADATEDROP1PART</b> keyword builds keys after splitting the date into the following parts:</p> <pre>YYYY-MM and MM-DD</pre> <ul style="list-style-type: none"> <li>- <b>FIRSTWORDRANGEORKEY.</b> Builds an additional key if the first word in the word stack is a skip word and the <b>NAMEFORMAT</b> keyword is set to <b>L</b>.</li> </ul>

Field	Description
	<ul style="list-style-type: none"> <li>- <b>BUILDPROBEDROPPFIRSTINIT.</b> Builds an additional probe for the words with a leading initial followed by at least two non-initial words. For example, the names MR M Dan Houting and Dan Houting match with the additional probe.</li> <li>- <b>EXTRACODEPROBES.</b> Builds an additional probe for each code with five or more characters in the word stack after formatting. The keys contain alphanumeric characters. Use this keyword to support model numbers for products that might not have digits.</li> </ul> <p><b>Note:</b> For more information about other valid keywords that the <code>ssan_get_ranges_encoded</code> control supports, see the NAMESET function keywords section in the <i>Informatica SSA-NAME3(EXTN) 10.2 Service Group Definition and Customization Guide</i>.</p>
DEBUG_ENABLE=[Y N]	Verifies the input parameters and writes debug information about the parameters to a specified log file. Default is N.
DEBUG_FILE	When <code>DEBUG_ENABLE</code> is set to Y, <code>DEBUG_FILE</code> specifies the name of the log file to which you want to write the debug information about the input parameters.
NONULLKEY=[Y N]	Optional. Specifies whether to include null keys in the SSA-NAME3 functions. When set to N, SSA-NAME3 returns a null key. When set to Y, SSA-NAME3 doesn't return a null key and returns 0 for the count of keys.

## ssan3\_convert\_ranges Controls

`Convert Ranges` does not require any Controls.

## ssan3\_match\_encoded Controls

The `ssan3_match_encoded` call compares two records and computes a match decision and a score. The `ssan3_match_encoded` controls set the purpose, the match level, and optionally, the layout of the search data and file data.

### Syntax

```

PURPOSE=<expression>
[MATCH_LEVEL=matchlevel] [+/-nn] [+/-nn]
[ADJWEIGHT=field,+/-n]
[NAMEFORMAT=name format]
[EARLYEXIT=Y/N]
[UNICODE_ENCODING=Unicode type]
[ENCODING=Unicode type]
[DELIMITER=delimiter]
[MATCH_OPTIONS=(fieldname:option,...)]
[MIN_MATCH_FIELDS=number of match field pairs]
[SEARCH=field1[(matching type expression1)],offset1,length1,... ,
fieldn[(matching type expressionn)],offsetn,lengthn]
[FILE=field1[(matching type expression1)],offset1,length1,... ,
fieldn[(matching type expressionn)],offsetn,lengthn]
[FMTOVERRIDE=Field1[(<Field ID>)],cX,cX,cX,cX:Field2[(<Field ID>)],cX,cX,cX,cX]
[LMM=Y/N/ONLY]
[LMM_FIELDS=field1,weight1[,... ,fieldn,weightn]]

```

```
[LWM_LIMIT=rejectscore[,acceptscore]]  
[EXT_MATCH_EXPLAIN=Y]  
[DEBUG_ENABLE=Y/N]  
[DEBUG_FILE=file name]
```

## Definitions

Control	Mandatory/Optional	Description
PURPOSE	Mandatory	<p>Specifies the name of the matching purpose to use in a match call. The PURPOSE control uses the following syntax:</p> <pre>PURPOSE=&lt;expression&gt;</pre> <p>Use one of the following formats for &lt;expression&gt;:</p> <pre>&lt;expression&gt; := &lt;Purpose_Name&gt; &lt;expression&gt; := &lt;Purpose_Name&gt;(&lt;Match_level&gt;) &lt;expression&gt; := (not&lt;expression&gt;) &lt;expression&gt; := (&lt;expression&gt; or &lt;expression&gt;) &lt;expression&gt; := (&lt;expression&gt; and &lt;expression&gt;) &lt;expression&gt; := (&lt;expression&gt;)</pre> <p>In the <code>ssan3_match</code> section of the SSA-NAME3 Workbench, under <b>Mandatory Controls</b>, you can find a list of the supported purpose names. With most standard populations, you can use one of the following predefined purposes:</p> <pre>Address Contact Division Fields Household Individual Organization Person_Name Resident Wide_Contact Geocode Filter1,Filter2,... Filter9</pre> <p>For more information about each purpose and its required and optional fields, from the <b>Help</b> menu of the SSA-NAME3 Workbench, click <b>Population Documentation</b>.</p> <p>The simple form of &lt;expression&gt; is <code>PURPOSE=&lt;Purpose_Name&gt;</code>.</p> <p>For example, <code>PURPOSE=Address</code> performs matching on the specified address fields.</p>
MATCH_LEVEL	Optional	<p>Controls the match level. With most standard populations, you can use one of the following predefined match levels:</p> <ul style="list-style-type: none"> <li>- Typical</li> <li>- Conservative</li> <li>- Loose</li> </ul> <p>The default value is Typical.</p> <p>Use the <code>MATCH_LEVEL</code> control to adjust the predefined accept or reject score limits that affect the match decisions.</p> <p>The <code>MATCH_LEVEL</code> control uses the following guidelines:</p> <ul style="list-style-type: none"> <li>- If one of the adjusted limits is greater than 100, the limit is set to 100.</li> <li>- If one of the adjusted limits is less than 0, the limit is set to 0.</li> <li>- If the adjusted accept limit is less than the reject limit, the accept limit is set to the reject limit.</li> <li>- If the accept match adjustment is 101, the accept limit is set to 101. As a result, SSA-NAME3 does not accept any record, and SSA-NAME3 either rejects a record or marks it as undecided.</li> </ul> <p>For example, if <code>MATCH_LEVEL=Loose+101-10</code>, the accept limit is set to 101.</p>

Control	Mandatory/Optional	Description
		<p>If a match call returns a score of 100, SSA-NAME3 marks the record as undecided.</p> <p>If a match call returns a score that is less than 100 and less than the reject limit, SSA-NAME3 rejects the record. If the score is less than 100 but greater than or equal to the reject limit, SSA-NAME3 marks the record as undecided.</p> <p><b>Note:</b> The MATCH_LEVEL control is not applicable to the Geocode purpose. If you specify the MATCH_LEVEL control with a Geocode purpose, SSA-NAME3 ignores the MATCH_LEVEL control.</p> <p>You can use one of the following match level expressions:</p> <p><b>matchlevel+/-nn</b></p> <p>Adjusts the accept and reject limits by using the same specified number. For example:</p> <pre>MATCH_LEVEL=Typical+20</pre> <p>If the predefined accept limit is 79 and reject limit is 50, the adjusted accept limit is 99 and reject limit is 70.</p> <p><b>matchlevel+/-nn+/-nn</b></p> <p>Adjusts the accept and reject limits by using the specified numbers. For example:</p> <pre>MATCH_LEVEL=Typical-40+40</pre> <p>If the predefined accept limit is 79 and reject limit is 50, the calculated accept limit is 39, which is less than the reject limit of 90. The adjusted accept limit is set to 90, and the reject limit remains at 90.</p>
ADJWEIGHT	Optional	<p>Adjusts the weight of a single field in a match purpose up or down relative to the other fields in the purpose. Use the following format:</p> <pre>ADJWEIGHT=field{+/-}n</pre> <p>The <code>field</code> is a valid field name in the purpose that you define in the PURPOSE= control, and <code>n</code> is a single-digit number.</p> <p><b>Note:</b> Standard populations use weights that are less than 10, so use smaller ADJWEIGHT adjustments. For example:</p> <pre>PURPOSE=Individual ADJWEIGHT=Person_Name+2</pre> <p>In this example, the definition increases the weight of the <code>Person_Name</code> field in the <code>Individual</code> match purpose by 2. The definition increases the importance of the <code>Person_Name</code> field and decreases the importance of the other fields. Experiment with different values and across a representative sample set of records to gauge the overall effect of this setting.</p>

Control	Mandatory/Optional	Description
NAMEFORMAT=L/R	Optional	<p>Defines whether the major word in the name or address is on the left end or the right end. For example, in Western person names, the family name is on the right end of the names.</p> <p>Use the <code>NAMEFORMAT</code> control to override the default name format. For more information about the default name format for a given standard population field and its effects, from the <b>Help</b> menu of SSA-NAME3 Workbench, click <b>Population Documentation</b>.</p> <p>You can provide extra weight to match the major words in the <code>Person_Name</code> field in some purposes such as <code>Person_Name</code>, <code>Household</code>, <code>Family</code>, and <code>Wide_Household</code>.</p> <p><b>Note:</b> The <code>NAMEFORMAT</code> control in a <code>match</code> call overrides the name format for all name and address fields in the match purpose, so use the <code>NAMEFORMAT</code> control with caution.</p>
EARLYEXIT=Y/N	Optional	<p>By default, SSA-NAME3 matching includes logic to check the score of the first field in a Purpose. If the score is low enough to cause a reject of the entire record, it takes an early exit from the match process. It results in a potential performance improvement, and the performance improves when it rejects more records. When an early exit is taken, the record score is set to 0. To preserve the real score of "unmatched" records (for example for analysis in the Developer's Workbench), you can disable the early exit logic by specifying <code>EARLYEXIT=N</code>.</p> <p>Use the <code>EARLYEXIT</code> control to handle the scores if the first Purpose field belongs to a Required or Best of group. If the first Purpose field belongs to a Required or Best of group and if the field score is less than the required limit for rejection of records, the early exit logic reduces the score to 0. If you do not want the match decisions to be based on the reduced scores for fields in Best of and Required groups, disable the early exit logic by specifying <code>EARLYEXIT=N</code>. If you specify multiple fields, the performance decreases because the early exit logic processes all the fields. For more information about Required and Best of groups, see the <i>Match Purposes</i> section in the <i>Identity Resolution Populations and Controls</i> guide.</p> <p><b>Note:</b> If you disable the early exit logic, the lightweight matching is disabled even though you have enabled the lightweight matching by configuring <code>LWM=Y</code>.</p>
UNICODE_ENCODING	Optional	Instructs SSA-NAME3 to accept Unicode data input, and specifies the Unicode format of the data that you want to pass.
ENCODING	Optional	Functions similar to the <code>UNICODE_ENCODING</code> keyword.
DELIMITER	Optional	Overrides the default delimiter - asterisk (*) if you pass the key field data by using the Tagged Data Format.

Control	Mandatory/Optional	Description
MATCH_OPTIONS	Optional	<p>Defines options for a specific field. The MATCH_OPTIONS control uses the following syntax:</p> <pre>MATCH_OPTIONS=(fieldname:option,...)</pre> <p>Use the MATCH_OPTIONS control for the following fields:</p> <p><b>Above</b></p> <p>Use the above option to set the match score to 100 when the match score exceeds the specified threshold value.</p> <p>The MATCH_OPTIONS control uses the following format for the above option:</p> <pre>MATCH_OPTIONS=(fieldname:above=&lt;value&gt;)</pre> <p>In the following example, the above option changes the match score that is above 70 to 100.</p> <pre>MATCH_OPTIONS=(Person_name:above=70)</pre> <p><b>Note:</b> You can't use the above and below options simultaneously in the MATCH_OPTIONS control.</p> <p><b>Below</b></p> <p>Use the below option to set the match score to 0 when the match score is below the specified threshold value.</p> <p>The MATCH_OPTIONS control uses the following format for the below option:</p> <pre>MATCH_OPTIONS=(fieldname:below=&lt;value&gt;)</pre> <p>In the following example, the below option changes the match score that is below 60 to 0.</p> <pre>MATCH_OPTIONS=(Person_name:below=60)</pre> <p><b>Note:</b> You can't use the above and below options simultaneously in the MATCH_OPTIONS control.</p> <p><b>Date</b></p> <p>The Date field accepts the Date_Format option. The MATCH_OPTIONS control uses the following format for the Date_Format option:</p> <pre>MATCH_OPTIONS=(DATE:Date_Format=&lt;DDMMYY&gt; &lt;MMDDYY&gt; &lt;YYMMDD&gt;,DATE:ts_error_penalty=&lt;number&gt;,DATE:ts_rescore=&lt;number&gt;,DATE:range=&lt;n&gt;,DATE:rangeopt=n, rangesel=n)</pre> <p>You can use the following additional date options:</p> <ul style="list-style-type: none"> <li>- ts_error_penalty. Deducts the specified penalty value from the match scores for transcription errors. For example, set MATCH_OPTIONS=(DATE:Date_Format=&lt;YYMMDD&gt;,DATE:ts_error_penalty=5). The penalty value to deduct from match score is 5. If the match score is 85, then final match score becomes 80. Default is 0.</li> <li>- ts_rescore. Interchanges the digits that are in the list of transcription errors to see if they match and then assigns a final score by deducting the penalty value from the highest score value.</li> </ul>

Control	Mandatory/Optional	Description
		<p>For example, you set  MATCH_OPTIONS=(DATE:Date_Format=&lt;YYMMDD&gt;,DATE:ts_error_penalty=5,DATE:ts_rescore=1). If you set the ts_rescore value to 1 or a greater number, and the transcription checks are for 0, 6 and 3, 5, 8. For the date is 78/12/10, the date match method performs a transcription check against 78/12/16, 75/12/10, and 73/12/10, and assigns a final score by deducting the penalty value.</p> <p>Default is 0.</p> <ul style="list-style-type: none"> <li>- range=&lt;n&gt;. Sets the range option for the Date field. For example, if you set  MATCH_OPTIONS=(DATE:Date_Format=&lt;YYMMDD&gt;,DATE:range=5), range = 5 is passed to the date matching method.  Default is 0.</li> <li>- rangeopt=n. Controls which date parts can participate in the range matching. This options allows range matching on the following parts of a date: <ul style="list-style-type: none"> <li>- 0 indicates all date parts.</li> <li>- 1 indicates year only.</li> <li>- 2 indicates month only.</li> <li>- 3 indicates day only.</li> </ul> Default is 0.</li> <li>- rangesel=n. Enables pattern selection for RANGEOPT. Use rangesel=1 to include complete date patterns, such as YYYY/MM/DD and YYYY/MM, and exclude partial date patterns, such as YYYY/M/D.  Default is 0.</li> </ul> <p><b>Geocode</b></p> <p>The Geocode field accepts the FORMAT and RADIUS options. The MATCH_OPTIONS control uses the following format for the FORMAT and RADIUS options:</p> <pre>MATCH_OPTIONS=(GEOCODE:FORMAT=1,GEOCODE:RADIUS=10000)</pre> <p>The FORMAT option indicates the order of the latitude and longitude coordinates that you specify.</p> <p>If GEOCODE:FORMAT=1, SSA-NAME3 considers the geographic coordinates in the following order:</p> <pre>*Geocode*&lt;Longitude&gt; &lt;Latitude&gt;***</pre> <p>If GEOCODE:FORMAT=0, SSA-NAME3 considers the geographic coordinates in the following order:</p> <pre>*Geocode*&lt;Latitude&gt; &lt;Longitude&gt;***</pre> <p>The default value is 0.</p> <p>The RADIUS option indicates the search radius to generate keys according to the latitude and longitude coordinates that you specify. The default value is 1000 m, and the default unit is meter.</p> <p>Use the following guidelines when you specify the geocode radius:</p> <ul style="list-style-type: none"> <li>- SSA-NAME3 considers the radius to be in meters even if you specify any other unit.</li> </ul>



Control	Mandatory/Optional	Description
		<p>- SSA-NAME3 ignores the decimal values and uses the whole number as the radius. For example, if you specify the radius as 850.8, SSA-NAME3 considers the radius as 850.</p> <p><b>Note:</b> SSA-NAME3 ignores any incorrect or unrecognised options in the MATCH_OPTIONS control and proceeds with matching, which might result in unexpected scores.</p> <p>SSA-NAME3 uses the following MATCH_OPTIONS to handle invalid Geocode values:</p> <ul style="list-style-type: none"> <li>- invnull=0 1. Set the value as 1 to enable SSA-NAME3 to consider the invalid Geocode values as null. Default is 0. The following example uses invnull=1: <pre> ssan3_match PURPOSE=Fields MATCH_OPTIONS=(Geocode:invnull=1) search=Geocode,0,20,Text,Geocode,22,21,Text file=Geocode,0,22,Text EXT_MATCH_EXPLAIN=Y 32.873398 -106.2882 131.873398 -106.2882 </pre> <p>In this example, the ssan3_info call uses the ITEM=match_report value and returns the weight as 0.</p> </li> <li>- exactm=0 1. Set the value as 1 to enable an exact match even when the Geocode values are invalid. Default is 0. The following example uses both the invnull=1 and exactm=0 1 options: <pre> ssan3_match PURPOSE=Fields MATCH_OPTIONS=(Geocode:invnull=1,Geocode:exactm=1) search=Geocode,0,20,Text,Geocode,22,22,Text file=Geocode,0,22,Text EXT_MATCH_EXPLAIN=Y 32.873398 -106.2882 </pre> <p>In this example, the ssan3_info call uses the ITEM=match_report value and returns both the score and weight as 100.</p> </li> </ul> <p><b>Address, Organization Name, Person Name</b></p> <p>When the Search or File field contains one token, this match option uses the field with more tokens for matching. This MATCH_OPTIONS control uses the following format:</p> <pre>MATCH_OPTIONS=(&lt;field&gt;:M C)</pre> <p>where field can be Address_Part1, Address_Part2, Organization_Name, and Person_Name. The option C applies to Address_Part1 and Address_Part2.</p> <p><b>Example:</b></p> <pre> MATCH_LEVEL=Conservative-20 PURPOSE=Person_Name MATCH_OPTIONS=(Person_Name:M) *Person_Name*James*** *Person_Name*James Andrews*** </pre> <p>With MATCH_OPTIONS=(Person_Name:M), *Person_Name*James Andrews*** is considered for matching.</p>

Control	Mandatory/Optional	Description
		<p>To provide the match scores for addresses based on the difference between two codes, use the following format:</p> <p>MATCH_OPTIONS=(Address_Part1:C)</p> <p>For example, 1215 MAIN ST versus 1225 MAIN ST has a higher match score than 1215 MAIN ST versus 1155 MAIN ST. The difference between codes 1215 and 1225 is 10, and the difference between 1215 and 1155 is 60.</p>
MIN_MATCH_FIELDS	Optional	<p>Specify the minimum number of field pairs that must be supplied. If a field is supplied for both the search or file record then it is counted.</p> <p>MIN_MATCH_FIELDS=number</p>

Control	Mandatory/Optional	Description
SEARCH and FILE	Optional	<p>Use the SEARCH and FILE controls to specify the field names and their offset and length pairs in the scatter or gather format for the data in the <i>Search Data</i> and <i>File Data</i> fields. If you do not specify the offset and length pairs for the field names, SSA-NAME3 considers the data in the tagged data format.</p> <p>You can extend any of the key fields and set the weight for the extended fields. The extended fields use the algorithm of the key fields. Use the extended fields to override the weight of the key fields in the run time. You can also specify the type of matching to perform between the data in the <i>Search Data</i> and <i>File Data</i> fields.</p> <p>Additionally, you can use the <i>above</i> or <i>below</i> options for the extended fields to set their match score to 100 or 0.</p> <p><b>Above</b></p> <p>Use the <i>above</i> option to set the match score to 100 when the match score exceeds the specified threshold value.</p> <p>The SEARCH and FILE controls use the following format for the <i>above</i> option:</p> <pre>SEARCH=&lt;Field Name&gt;(&lt;Field ID&gt;;above:&lt;value&gt;),&lt;offset&gt;,&lt;length&gt; FILE=&lt;Field Name&gt;,&lt;offset&gt;,&lt;length&gt;</pre> <p>In the following example, the <i>above</i> option changes the match score that is above 70 to 100.</p> <pre>SEARCH=Person_name(0;above:70),0,4 FILE=Person_name,0,4</pre> <p><b>Note:</b> You can't use the <i>above</i> and <i>below</i> options simultaneously in the SEARCH and FILE controls.</p> <p><b>Below</b></p> <p>Use the <i>below</i> option to set the match score to 0 when the match score is below the specified threshold value.</p> <p>The SEARCH and FILE controls use the following format for the <i>below</i> option:</p> <pre>SEARCH=&lt;Field Name&gt;(&lt;Field ID&gt;;below:&lt;value&gt;),&lt;offset&gt;,&lt;length&gt; FILE=&lt;Field Name&gt;,&lt;offset&gt;,&lt;length&gt;</pre> <p>In the following example, the <i>below</i> option changes the match score that is below 60 to 0.</p> <pre>SEARCH=Person_name(0;below=60),0,4 FILE=Person_name,0,4</pre> <p><b>Note:</b> You can't use the <i>above</i> and <i>below</i> options simultaneously in the SEARCH and FILE controls.</p> <p>You can use the following matching types:</p> <ul style="list-style-type: none"> <li>- Exact matching. Returns 100% score only if the search data values match with the file data values.</li> <li>- Fuzzy matching. Returns 0 to 100% score based on how close the search data and file data values match.</li> <li>- Inexact matching. Returns 100% score if the search data values do not match with the file data values.</li> <li>- Range matching. Returns 100% score if the search data and file data values are within the specified range.</li> </ul>

Control	Mandatory/Optional	Description
		<p><b>Note:</b> If you specify different matching types in the SEARCH and FILE parameters, the matching type that you specify in the SEARCH parameter takes precedence.</p> <p><b>Exact Matching</b></p> <p>Use the following format to perform exact matching:</p> <pre>PURPOSE=&lt;Purpose Name&gt; SEARCH=&lt;Field Name&gt;[(&lt;Field ID&gt;;&lt;Weight&gt;;Exact[:B Z ZB])],&lt;Offset&gt;,&lt;Length&gt; FILE=[(&lt;Field ID&gt;;&lt;Weight&gt;;Exact)],&lt;Offset&gt;,&lt;Length&gt;</pre> <p>If you want to extend a key field, use any number as the Field ID value. If you do not want to extend any key field, use 0 as the Field ID value. If you extend a key field in the search or file data, you must extend the corresponding key field in the search or file data.</p> <p>The value B indicates a null value, the value Z indicates a zero value, and the value ZB indicates a null or zero value.</p> <p>Use the following format to perform exact matching when you specify multiple fields in the PURPOSE:</p> <pre>PURPOSE=&lt;Purpose Name&gt; SEARCH=&lt;Field Name&gt;([[!]&lt;field_number&gt;[;weight]][;][qualifier:from;to][;PurposeName]) FILE=&lt;Field Name&gt;([[!]&lt;field_number&gt;[;weight]][;][qualifier:from;to][;PurposeName])</pre> <p>Use the [;PurposeName] option preceded by three semi-colons to indicate that the field should be restricted only to the purpose specified in this option.</p> <p>The following is a sample expression which does not use the [;PurposeName] option:</p> <pre>Controls: 'MATCH_LEVEL=Typical Purpose=(Person_Name OR Address) ' Search data: '*Person_Name*Mick*Address_Part1*Torrens***' File data: '*Person_Name*Mick*Address_Part1*Braddon***'</pre> <p>The preceding example returns a score of 80% since Address_Part1 fields do not match and reduces the score returned by the Person_Name purpose. To restrict the Address_Part1 field to the Address purpose, use the [;PurposeName] option as shown in the following sample expression which returns a score of 100%:</p> <pre>Controls: 'MATCH_LEVEL=Typical Purpose=(Person_Name OR Address) ' Search data: '*Person_Name*Mick*Address_Part1(;;;Address)*Torrens***' File data: '*Person_Name*Mick*Address_Part1(;;;Address)*Braddon***'</pre> <p>The (;;;Address) option used in the above example indicates that the Address_Part1 field should be restricted only to the Address purpose and hence returns a score of 100%.</p> <p>The following sample expressions perform exact matching between the search data and file data values:</p> <ul style="list-style-type: none"> <li>- PURPOSE=Person_Name SEARCH=Person_Name(Exact),0,16 FILE=Person_Name,0,16. The expression returns 100% score only if the search data value exactly matches with the file data value.</li> <li>- PURPOSE=Fields SEARCH=Person_Name(2;5;Exact),0,16 FILE=Person_Name(2;Exact),0,16. The expression creates an extended field for the Person_Name field named Person_Name(2) and sets the weight for the extended field to 5. The search data value returns 100% score if the search data value exactly matches with the file data value.</li> </ul>

Control	Mandatory/Optional	Description
		<ul style="list-style-type: none"> <li>- PURPOSE=Person_Name SEARCH=Person_Name(0;5;Exact),0,16 FILE=Person_Name,0,16. The expression sets the weight of the Person_Name field to 5 without creating any extended field. The search data value returns 100% score if the search data value exactly matches with the file data value.</li> <li>- PURPOSE=Person_Name SEARCH=Person_Name(1;Exact),0,16 FILE=Person_Name(1;Exact),0,16. The expression creates an extended field for the Person_Name field named Person_Name(1). The extended field uses the weight of the Person_Name field. The search data value returns 100% score if the search data value exactly matches with the file data value.</li> <li>- PURPOSE=Fields SEARCH=Date,0,8 FILE=Date(Exact:Z),0,8. The expression returns 100% score if the file data value is 0.</li> <li>- PURPOSE=(Person_name OR Address) SEARCH=Person_Name,0,50,Address_Part1(;;Address). The Address_Part1 is an optional field in the Person_name purpose. The (;;Address) option indicates that the Address_Part1 should be used only by the Address purpose.</li> </ul> <p><b>Reverse Matching</b></p> <p>Reverses the original score. Use the following format to perform reverse matching:</p> <pre>PURPOSE=&lt;Purpose Name&gt; SEARCH=&lt;Field Name&gt;[(&lt;!Field ID&gt;;&lt;Weight&gt;;)],&lt;Offset&gt;,&lt;Length&gt; FILE=&lt;Field Name&gt;[(&lt;!Field ID&gt;;&lt;Weight&gt;;)],&lt;Offset&gt;,&lt;Length&gt;</pre> <p>The exclamation mark symbol (!) symbol preceding the numerical value for Field ID triggers score reversal. The exclamation mark symbol (!) is mandatory for the SEARCH field and optional for the FILE field.</p> <p>The following sample expression performs reverse matching of the organization name:</p> <pre>PURPOSE=Fields SEARCH=Organization_Name(!0),0,11,Person_name,11,14 FILE=Organization_Name(!0),0,11,Person_name,11,14 InformaticaJohn Smith InformaticaJohn Smith</pre> <p>For the preceding sample expression, you get a reverse score of 0% for Informatica instead of the usual score of 100%.</p> <p><b>Fuzzy Matching</b></p> <p>Use the following format to perform fuzzy matching:</p> <pre>SEARCH=&lt;Field Name&gt;[(&lt;Field ID&gt;;&lt;Algorithm Name&gt;[:&lt;Upper Score Limit&gt;;&lt;Lower Score Limit&gt;] FILE=&lt;Field Name&gt;[(&lt;Field ID&gt;;&lt;Algorithm Name&gt;[:&lt;Upper Score Limit&gt;;&lt;Lower Score Limit&gt;]</pre> <p>The format uses the following parameters:</p> <ul style="list-style-type: none"> <li>- Field Name. Name of the SSA-NAME3 field.</li> <li>- Field ID. Number that you want to assign to the extended field. If you want to extend a key field, use any number as the Field ID value. If you do not want to extend a key field, use 0 as the Field ID value. If you extend a key field in the search data, extend the corresponding key field in the file data.</li> <li>- Algorithm Name. Name of the supported algorithm that performs the fuzzy matching.</li> </ul>

Control	Mandatory/Optional	Description
		<p>Use one of the following values:</p> <ul style="list-style-type: none"> <li>- Levenshtein. Scores the records based on the minimum number of single-character edits required to change one word into the other.</li> <li>- Dice. Scores the records based on the Dice's coefficient.</li> <li>- JaroWinkler. Scores the records based on the minimum number of single-character transpositions required to change one word into the other.</li> </ul> <p>- Upper Score Limit/Lower Score Limit. Minimum score required for the records to be considered as matches or unique records.</p> <p>If a score is more than the upper score limit, the records are considered as a match. If a score is less than the lower score limit, the records are considered as unique records. If a score is between the upper and lower score limits, the records are considered as undecided matches. Default is 0.</p> <p>The following sample expressions perform fuzzy matching between the search data and file data values:</p> <ul style="list-style-type: none"> <li>- SEARCH=Attribute1(1;Levenshtein:79;60) FILE=Attribute1(1;Levenshtein:79;60). The expression extends the Attribute1 field and uses the Levenshtein algorithm to perform fuzzy matching.</li> <li>- SEARCH=Attribute1(0;JaroWinkler) FILE=Attribute1(0;JaroWinkler). The expression does not extend the Attribute1 field and uses the JaroWinkler algorithm to perform fuzzy matching.</li> </ul> <p><b>Inexact Matching</b></p> <p>Use the following format to perform inexact matching:</p> <pre>PURPOSE=&lt;Purpose Name&gt; SEARCH=&lt;Field Name&gt;[(&lt;Field ID&gt;;&lt;Weight&gt;;!Exact[:B Z ZB])],&lt;Offset&gt;,&lt;Length&gt; FILE=[(&lt;Field ID&gt;;&lt;Weight&gt;;!Exact)],&lt;Offset&gt;,&lt;Length&gt;</pre> <p>If you want to extend a key field, use any number as the Field ID value. If you do not want to extend any key field, use 0 as the Field ID value. If you extend a key field in the search or file data, you must extend the corresponding key field in the search or file data.</p> <p>The value B indicates a null value, the value Z indicates a zero value, and the value ZB indicates a null or zero value.</p> <p>The following sample expressions perform inexact matching between the search data and file data values:</p> <ul style="list-style-type: none"> <li>- PURPOSE=Person_Name SEARCH=Person_Name(!Exact),0,16 FILE=Person_Name,0,16. The expression returns 100% score if the search data value does not match with the file data value.</li> <li>- PURPOSE=Fields SEARCH=Person_Name(1;5;!Exact),0,16 FILE=Person_Name(1;!Exact),0,16. The expression creates an extended field for the Person_Name field named Person_Name(1) and sets the weight for the extended field to 5. The search data value returns 100% score if the search data value does not match with the file data value.</li> <li>- PURPOSE=Person_Name SEARCH=Person_Name(0;5;!Exact),0,16 FILE=Person_Name,0,16. The expression sets the weight of the Person_Name field to 5 without creating any extended field. The search data value returns 100% score if the search data value does not match with the file data value.</li> <li>- PURPOSE=Fields SEARCH=Person_Name(3;!Exact),0,16 FILE=Person_Name(3;!Exact),0,16. The expression creates an extended field for the Person_Name field named Person_Name(3), and the extended field uses the weight of the Person_Name field. The search data value returns 100% score if the search data value does not match with the file data value.</li> </ul>

Control	Mandatory/Optional	Description
		<p>- PURPOSE=Fields SEARCH=Date,0,8 FILE=Date(!Exact:ZB),0,8. The expression returns 100% score if the file data value is not 0 or null.</p> <p><b>Range Matching</b></p> <p>Use one of the following formats to perform range matching on numeric fields:</p> <p>- PURPOSE=&lt;Purpose Name&gt; SEARCH=&lt;Field Name&gt;[(&lt;Field ID&gt;;&lt;Weight&gt;;Range:&lt;Negative Limit&gt;;&lt;Positive Limit&gt;)],&lt;Offset&gt;,&lt;Length&gt; FILE=&lt;Field Name&gt;[(&lt;Field ID&gt;;&lt;Weight&gt;;Range:&lt;Negative Limit&gt;;&lt;Positive Limit&gt;)],&lt;Offset&gt;,&lt;Length&gt;</p> <p>If you want to extend a key field, use any number as the Field ID value. If you do not want to extend any key field, use 0 as the Field ID value. If you extend a key field in the search or file data, you must extend the corresponding key field in the search or file data.</p> <p>For example, the PURPOSE=Person_Name SEARCH=Person_Name,0,16,Attribute1(1;5;Range:20;10),16,2 FILE=Person_Name,0,16,Attribute1(1;Range),16,2 expression creates an extended field for the Attribute1 field named Attribute1(1) and sets the weight for the extended field to 5. If the Attribute1(1) value in the search data is 100, the range for the search data is 100-20 to 100+10, which is 80 to 110. The range matching returns 100% score if the Attribute1(1) value in the file data is within the search data range.</p> <p>If both the positive and negative limits are the same, you can specify the range in the (Range:&lt;Limit&gt;) format. For example, (Range:20) equals (Range:20;20) .</p> <p>- PURPOSE=&lt;Purpose Name&gt; SEARCH=&lt;Field Name&gt;[(&lt;Field ID&gt;;&lt;Weight&gt;;Range:%&lt;Negative Limit&gt;;&lt;Positive Limit&gt;)],&lt;Offset&gt;,&lt;Length&gt; SEARCH=&lt;Field Name&gt;[(&lt;Field ID&gt;;&lt;Weight&gt;;Range:%&lt;Negative Limit&gt;;&lt;Positive Limit&gt;)],&lt;Offset&gt;,&lt;Length&gt;</p> <p>If you want to extend a key field, use any number as the Field ID value. If you do not want to extend any key field, use 0 as the Field ID value. If you extend a key field in the search or file data, you must extend the corresponding key field in the search or file data.</p> <p>For example, the PURPOSE=Person_Name SEARCH=Person_Name,0,16,Attribute1,16,2 FILE=Person_Name,0,16,Attribute1(Range:%10;20),16,2 expression indicates that if the Attribute1 value in the file data is 100, the range for the file data is 100-10% of the Attribute1 value to 100+20% of the Attribute1 value, which is 90 to 120. The range matching returns 100% score if the Attribute1 value in the search data is within the file data range.</p> <p>If both the positive and negative limits are the same, you can specify the range in the (Range:%&lt;Limit&gt;) format. For example, (Range:%20) equals (Range:%20;20) .</p> <p>- PURPOSE=&lt;Purpose Name&gt; SEARCH=&lt;Field Name&gt;[(Range:F S)],&lt;Offset&gt;,&lt;Length&gt; FILE=&lt;Field Name&gt;[(Range:F S)],&lt;Offset&gt;,&lt;Length&gt;</p> <p>The value F indicates to use the file data to create the range, and the value S indicates to use the search data to create the range.</p> <p>For example, the PURPOSE=Fields SEARCH=Date(Range:F),0,4 FILE=Date(Range:F),0,4,Date(Range:F),4,4 expression indicates that if the file data is 50008000, the range for the search data is 5000 to</p>

Control	Mandatory/Optional	Description
		<p>8000. The range matching returns 100% score if the Date value in the search data is within the file data range.</p> <p>Use one of the following formats to perform range matching on date fields:</p> <ul style="list-style-type: none"> <li>- PURPOSE=&lt;Purpose Name&gt; SEARCH=&lt;Date Field Name&gt;[(Range:F S)],&lt;Offset&gt;,&lt;Length&gt; FILE=&lt;Date Field Name&gt;[(Range:F S)],&lt;Offset&gt;,&lt;Length&gt;</li> </ul> <p>The value F indicates to use the file data to create the range, and the value S indicates to use the search data to create the range.</p> <p>For example, the PURPOSE=Fields SEARCH=Date(Range:F),0,8 FILE=Date(Range:F),0,8,Date(Range:F),8,8 expression indicates that if the file data is 1999101019991020, the range for the search data is 19991010 to 19991020. The range matching returns 100% score if the Date value in the search data is within the file data range.</p> <ul style="list-style-type: none"> <li>- PURPOSE=&lt;Purpose Name&gt; SEARCH=&lt;Date Field Name&gt;[(DateRange:&lt;Negative Limit&gt;;&lt;Positive Limit&gt;)],&lt;Offset&gt;,&lt;Length&gt; SEARCH=&lt;Date Field Name&gt;[(DateRange:&lt;Negative Limit&gt;;&lt;Positive Limit&gt;)],&lt;Offset&gt;,&lt;Length&gt;</li> </ul> <p>For example, the PURPOSE=Fields SEARCH=Date(Range:F),0,8 FILE=Date(DateRange:5;10),8,8 expression indicates that if the Date value in the file data is 20150410, the range for the file data is 20150405 to 20150420. The range matching returns 100% score if the Date value in the search data is within the file data range.</p> <p>If both the positive and negative limits are the same, you can specify the range in the (DateRange:&lt;Limit&gt;) format. For example, (DateRange:7) equals (DateRange:7;7) .</p> <ul style="list-style-type: none"> <li>- PURPOSE=&lt;Purpose Name&gt; SEARCH=&lt;Date Field Name&gt;[(DateRange:%&lt;Negative Limit&gt;;&lt;Positive Limit&gt;)],&lt;Offset&gt;,&lt;Length&gt; SEARCH=&lt;Date Field Name&gt;[(DateRange:%&lt;Negative Limit&gt;;&lt;Positive Limit&gt;)],&lt;Offset&gt;,&lt;Length&gt;</li> </ul> <p>For example, the PURPOSE=Fields SEARCH=Date,0,8 FILE=Date(DateRange:%10;20),0,8 expression indicates that if the Date value in the file data is 19991010, the range for the file data is 10-10% of the Date value to 10+20% of the Date value, which is 19991009 to 19991012. The range matching returns 100% score if the Date value in the search data is within the file data range.</p> <p>If both the positive and negative limits are the same, you can specify the range in the (DateRange:%&lt;Limit&gt;) format. For example, (DateRange:%20) equals (DateRange:%20;20) .</p>



Control	Mandatory/Optional	Description
FMTOVERRIDES	Optional	<p>Specifies whether to disable or override a list of category names and category types.</p> <p>Use the following format:</p> <pre>PURPOSE=&lt;Purpose Name&gt; SEARCH=&lt;Field Name&gt;[(&lt;Field ID&gt;;&lt;Weight&gt;;)],&lt;Offset&gt;,&lt;Length&gt; FILE=&lt;Field Name&gt;[(&lt;Field ID&gt;;&lt;Weight&gt;;)],&lt;Offset&gt;,&lt;Length&gt; FMTOVERRIDES=&lt;Field Name1&gt;[(&lt;FieldID&gt;)],cX,cX,cX,cX:&lt;Field Name2&gt;[(&lt;Field ID&gt;)],cX,cX,cX,cX]</pre> <p>The format uses the following values:</p> <ul style="list-style-type: none"> <li>- c. Use one of the following values: <ul style="list-style-type: none"> <li>- N - Disables the specified edit-list category names during key generation process.</li> <li>- T - Disables the specified edit-list category types during key generation process.</li> <li>- X - Overrides the specified edit-list category type with another category type.</li> <li>- Y - Overrides the specified edit-list category name with another category type.</li> </ul> </li> <li>- X. Denotes the category name or the category type in the edit-list that you want to disable or override.</li> </ul> <p>The following sample configuration disables nick name words:</p> <pre>PURPOSE=Fields SEARCH=Person_name,0,4 FILE=Person_name,0,4 FMTOVERRIDES=Person_Name(1),NRRNNNK,TS:Person_Name(0),NRRNNK K Mick Mike</pre> <p>For the preceding sample configuration, you get a score of 75% as SSA-NAME3 disables the NK category name in the edit-list.</p>

Control	Mandatory/Optional	Description
FILTER_SEARCHVALUES	Optional	<p>Use the FILTER_SEARCHVALUES control to specify a list of values to match with the data in the Search Data field, File Data field, or both the fields. Use the Filter purpose to specify the data in the Search Data and File Data fields.</p> <p>The FILTER_SEARCHVALUES control uses the following format to specify a list of search values:</p> <pre>PURPOSE=(Filter&lt;Filter Purpose Number&gt;,...) FILTER_SEARCHVALUES=&lt;Filter Purpose Number&gt;,&lt;List of Flags&gt;, (&lt;List of Values&gt; {&lt;Value&gt;,&lt;List of Values&gt;})</pre> <p>For example:</p> <pre>PURPOSE=(Filter2 AND Filter3) FILTER_SEARCHVALUES=2,SP, (WILLIAM,BILL),3,FP,{ROBERT,(ROB,ROBBIE)}</pre> <p>The FILTER_SEARCHVALUES control uses the following parameters:</p> <p><b>Filter Purpose Number</b></p> <p>Indicates the Filter purpose number for which you define a list of values.</p> <p>For example, if you use the Filter5 purpose to specify the search data, use 5 as the Filter purpose number.</p> <p><b>(List of Values) (Value,(List of Values))</b></p> <p>Indicates the list of values to match with the search data, file data, or both. You can specify a list of values, or you can pair a value with a list of values.</p> <p>For example:</p> <pre>FILTER_SEARCHVALUES=2,SP,(WILLIAM,BILL),3,FP,{ROBERT, (ROB,ROBBIE)}</pre> <p>Use appropriate flags to indicate whether to match the specified values with the search data, file data, or both.</p> <p><b>List of Flags</b></p> <p>Indicates whether to match the specified values with the search data, file data, or both. You can use multiple flags. For example, FP.</p> <p>If you specify a list of values, you can use the following flags:</p> <ul style="list-style-type: none"> <li>- S. Indicates to match the values with the search data.</li> <li>- F. Indicates to match the values with the file data.</li> <li>- B. Indicates to match the values with both the search and file data.</li> <li>- P. Indicates to match the values in the list with the search data, file data, or both based on the length of the values in the list. This flag functions in conjunction with other flags.</li> </ul> <p>For example, specify FILTER_SEARCHVALUES=2,SP,Rob as one of the SCORE-LOGIC controls and *Filter2*Rose*** in the search data. The length of Rob is three characters, so the matching considers only the first three characters of the search data. The matching does not return 100% score because Rob in the list does not exactly match with the first three characters of the search data, which are Ros.</p> <p>If you pair a value with a list of values, you can use the following flags:</p> <ul style="list-style-type: none"> <li>- S. Indicates to match the value with the search data and the list of values with the file data.</li> </ul>

Control	Mandatory/Optional	Description
		<ul style="list-style-type: none"> <li>- F. Indicates to match the value with the file data and the list of values with the search data.</li> <li>- P. Indicates to match the values in the list with the search data, the file data, or both based on the length of the values in the list. This flag functions in conjunction with other flags.</li> </ul> <p>For example, the <code>FILTER_SEARCHVALUES=3,S,{ROBERT,(ROB,ROBBIE)}</code> control indicates to match Robert with the search data and the list of values, Rob and Robbie, with the file data.</p> <p>You can also perform inexact matching to get 100% score if the list values do not match with the search or file data.</p> <p>To perform inexact matching, use the following format when you define the Filter purpose:</p> <pre>PURPOSE=(NOT Filter&lt;Number&gt;, NOT Filter&lt;Number&gt;, ...)</pre> <p>The usage of NOT indicates that you want to perform inexact matching. For example, the <code>PURPOSE=(NOT Filter2) FILTER_SEARCHVALUES=2,SP,(WILLIAM,BILL)</code> expression returns 100% score if the list values do not match with the search data.</p>
LWM=Y/N/ONLY	Optional	<p>Enables or disables lightweight matching. Use the value <code>Y</code> to enable lightweight matching. Lightweight matching uses a fast score estimate to reject the obvious mismatches. The records that lightweight matching passes go to the full scoring for robust scoring and ranking. SSA-NAME3 returns the full score and the decision to the caller.</p> <p><b>Note:</b> If you create system definition files by using the SDF Wizard, the lightweight matching is enabled by default.</p> <p>Use the value <code>N</code> to disable lightweight matching. SSA-NAME3 matching performs full scoring on all the matching records.</p> <p>Use the value <code>ONLY</code> to enable lightweight matching and disable full scoring. Lightweight matching returns the estimate as the final score to the caller.</p>
LWM_FIELDS	Optional	<p>Specifies the fields to which you want to apply lightweight matching and their weights. These values override the values that you have defined in the match purpose during the run time. Based on the lightweight matching scores, SSA-NAME3 rejects the obvious mismatches. If you do not set any value, SSA-NAME3 retrieves the fields from the match purpose and assigns equal weight to them.</p> <p>The syntax of the LWM_FIELDS control is as follows:</p> <pre>LWM_FIELDS=&lt;field1&gt;,&lt;weight1&gt;[,...,&lt;fieldn&gt;,&lt;weightn&gt;]</pre> <p>where <code>field</code> is a valid field name that you have defined in the Purpose control, and <code>weight</code> is the relative significance of the specified field (0-100) when compared to the other fields.</p> <p>For example, <code>LWM_FIELDS=Person_Name,5,Address_Part1,1</code></p> <p>Lightweight matching is useful when you apply it to the fields that have low variations such as addresses. Lightweight matching is not efficient for the fields with high variations, where SSA-NAME3 handles the variations through Edit-list, and lightweight matching might incorrectly reject the records.</p>

Control	Mandatory/Optional	Description
LWM_LIMIT	Optional	<p>Specifies the accept and reject limits for the lightweight matching score. Based on the limits, SSA-NAME3 accepts or rejects the search results.</p> <p>The syntax of the LWM_LIMIT control is as follows:  <code>LWM_LIMIT=&lt;Reject&gt;[,&lt;Accept&gt;]</code></p> <p>where <code>Reject</code> and <code>Accept</code> are the integer values ranging from 0 through 100.</p> <p>For example, <code>LWM_LIMIT=50,90</code></p> <p>If <code>LWM=N</code>, the LWM_LIMIT control has no effect.</p> <p>If <code>LWM=Y</code>, SSA-NAME3 rejects the lightweight matching scores that are less than the reject limit. The accept limit has no effect, and you can omit it.</p> <p>If <code>LWM=ONLY</code>, SSA-NAME3 rejects the lightweight matching scores that are less than the reject limit. It accepts the scores that are greater than the accept limit. It marks the scores of the records that are greater than or equal to the reject limit and less than the accept limit as undecided.</p> <p>The default reject limit is 65, and the default accept limit is 90. If you have not set the accept limit and the reject limit is greater than 90, the accept limit is equal to the reject limit.</p>
EXT_MATCH_EXPLAIN=Y	Optional	<p>Adds extra information about matching to the details that the <code>ssan3_info</code> call returns using the <code>ITEM=match_report</code> value.</p>
DEBUG_ENABLE=[Y N]	Optional	<p>Verifies the input parameters and writes debug information about the parameters to a specified log file.</p> <p>Default is N.</p>
DEBUG_FILE	Optional	<p>When <code>DEBUG_ENABLE</code> is set to Y, <code>DEBUG_FILE</code> specifies the name of the log file to which you want to write the debug information about the input parameters.</p>
OVERRIDE_FIELDS	Optional	<p>Overrides the match score of extended fields based on the options that you specify for each field.</p> <p>Use the following syntax for the <code>OVERRIDE_FIELDS</code> control:  <code>OVERRIDE_FIELDS=field-name (Option) , field-name (Option)</code></p> <p>For example, if the search data and the field data values for the code field return a match score of 80%, you can override the match score to 100% with the following sample <code>OVERRIDE_FIELDS</code> control:  <code>OVERRIDE_FIELDS=Code (Range:%7)</code></p> <p>To override the match score to 100%, you can add the range based on the difference between the values of the code fields in the search data and the field data.</p> <p>To apply the <code>OVERRIDE_FIELDS</code> control to multiple fields, use the following sample format:  <code>OVERRIDE_FIELDS=Code (Range:%5) , Product_name (Range:%10) , Attribute1 (Range:%5)</code></p>

# ssan3\_get\_word\_stack

The **ssan3\_get\_word\_stack** call returns the word stack for all the fields. The **ssan3\_get\_word\_stack** call splits multiple names into separate calls and appends each word stack to create a final combined word stack.

## Syntax

```
FIELD=field name  
[KEY_LEVEL=key level]  
[STB_WORDS_ONLY=Y/N]  
[STB_WORDS_INLINE=Y/N]  
[STB_WORDS_NODUP=Y/N]  
[LAYOUT=[field1,]offset1,length1,... , [fieldn,]offsetn,lengthn]
```

## Definitions

Field	Mandatory/Optional	Description
FIELD	Mandatory	Defines the field for which you want to build keys. With most standard populations, you can use one of the following predefined fields: <ul style="list-style-type: none"><li>- Person_Name</li><li>- Organization_Name</li><li>- Address_Part1</li><li>- Geocode</li></ul> You can use only one key field in a function call. If you want to build a key on another field that contains a different type of data, use two separate functions so that you can store the keys in two separate indexes.
KEY_LEVEL	Optional	Indicates the type of key level that you want to build. With most standard populations, you can use one of the following predefined values: <ul style="list-style-type: none"><li>- Standard</li><li>- Extended</li><li>- Limited</li></ul> Default is Standard.
STB_WORDS_ONLY=Y/N	Optional	Specifies to return only stabilized words for an input string. Default is N.
STB_WORDS_INLINE=Y/N	Optional	Specifies to return stabilized words in a single line for an input string. Default is N.

Field	Mandatory/Optional	Description
STB_WORDS_NODUP=Y/N	Optional	Specifies to return stabilized words without duplicates for an input string. Default is N.
LAYOUT	Optional	Specifies the offset and length pairs of the data in the Scatter / Gather format to be passed for key building in the <code>Key Field Data</code> parameter. For more information about the <code>Key Field Data</code> parameter, see <a href="#">“Common Parameters” on page 23</a> . If you do not specify the layout, the output data appears in the Tagged Data format.

## ssan3\_info Controls

The `ssan3_info` call returns information about a selected system or identifies the population file to use for the specified sample data.

### Syntax

The `ssan3_info` control uses the following format:

```
ITEM=<Item Type> SELECT=<Field Name|Purpose Name>|ITEM_DATA=utf-8,<Length in Bytes>,<Sample Data>|DEBUG_ENABLE=<Y|N>|DEBUG_FILE=<File Name>
```

## Definitions

Control	Description
ITEM	<p>Specifies the type of information to retrieve. Use one of the following item types:</p> <ul style="list-style-type: none"> <li>- <code>item</code>. Retrieves all the supported item types that you can use.</li> <li>- <code>analyze_match</code>. Analyzes the SSA-NAME3 fields that returns low match score and recommends how to improve the score.</li> <li>- <code>column_check</code>. Checks for concatenated columns.</li> <li>- <code>ext_field_types</code>. Retrieves the details of the fields that you extend.</li> <li>- <code>field_id</code>. Retrieves the internal ID of a field.</li> <li>- <code>key_level</code>. Retrieves all the available values of the <code>Key_Level</code> control.</li> <li>- <code>map_data</code>. Recommends the SSA-NAME3 field that you can use for the specified sample data.</li> <li>- <code>map_field</code>. Recommends the SSA-NAME3 field that matches with the column name in the sample data set.</li> <li>- <code>map_profile</code>. Recommends the columns that you can combine.</li> <li>- <code>map_population</code>. Recommends a suitable population that has most rules matching with the specified sample data.</li> <li>- <code>match_report</code>. Provides a match explainability report. To get the report, run the <code>ssan3_match</code> call. If you set the <code>EXT_MATCH_EXPLAIN</code> control to Y, it enables the additional match explainability information.</li> <li>- <code>purpose.detail</code>. Generates an XML report that contains the details of the match purposes. By default, you get the details of all the match purposes. If you specify a purpose name in the <code>SELECT CONTROL</code>, you get the details of the specified purpose.</li> <li>- <code>purpose.list</code>. Retrieves the details of the field that you specify in the <code>SELECT</code> control.</li> <li>- <code>results.name_in</code>. Retrieves the input data that you specify in the <code>Key Field Data</code> field.</li> <li>- <code>results.categories</code>. Retrieves the Edit-list categories triggered during the previous call.</li> <li>- <code>results.cleaned</code>. Displays the input data after cleansing the data.</li> <li>- <code>results.word_stack</code>. Retrieves the word-stack items triggered during the previous call.</li> <li>- <code>results.key-table</code>. Retrieves the additional data that the <code>KEY-TABLE</code> keyword generates. This additional data explains how SSA-NAME3 constructs each key based on the word stack order.</li> <li>- <code>results.search_table</code>. Retrieves the full search table, including the search limit values.</li> <li>- <code>results.stab</code>. Retrieves the additional data that the <code>DEBUG</code> keyword generates.</li> </ul> <p>The following sample response indicates how the key is generated for JAMES ALAN MACKINTOSH:  <code>0 'UU?BGI\$\$ UU?BGI\$/ 60 0000 20 A1 P 00 03+01+00+00'</code></p> <p>The value <code>03+01+00+00</code> indicates that the key is generated in the order, MACKINTOSH JAMES.</p> <ul style="list-style-type: none"> <li>- <code>suggest_population</code>. Recommends a population based on the character code points present in the data.</li> <li>- <code>trace_word_stack</code>. Retrieves the <code>TRACE Service</code> details. For more information about the <code>TRACE Service</code>, see the <code>TRACE</code> chapter in the <i>Informatica SSA-NAME3 (EXTN) Service Group Application Reference</i> guide.</li> </ul> <p>The following sample uses the <code>trace_word_stack</code> control to retrieve the <code>TRACE Service</code> details:</p> <pre>Controls: 'ITEM=trace_word_stack SELECT=Person_name ITEM_DATA=utf-8,1,14,michael taylor'</pre>

Control	Description
	<pre>Rsp_code: '0' SSA_msg: '' Count: 2 0 'MICHAEL YYCC000006 2036 1 'TAYLOR MYCU008013 3000</pre>
SELECT	<p>Specifies the field name or the purpose name based on the item type to retrieve information about the field or purpose.</p> <p>The <b>SELECT</b> control is applicable to the following item types:</p> <ul style="list-style-type: none"> <li>- purpose.detail</li> <li>- purpose.list</li> <li>- fieldlen</li> <li>- match_explain_count</li> <li>- field.options</li> </ul> <p>The <code>purpose.list</code> and <code>fieldlen</code> item types require the <b>SELECT</b> control.</p> <p>The <b>SELECT</b> control is optional for the <code>field.options</code>, <code>match_explain_count</code>, and <code>purpose.detail</code> item types. If you don't specify the <b>SELECT</b> control for these item types, the <code>ssan3_info</code> call retrieves information about all the fields or purposes.</p> <p>For example:</p> <p><code>ITEM=fieldlen SELECT=CreditCard</code> returns the key length of the <code>CreditCard</code> field and <code>ITEM=purpose.detail SELECT=Person_Name</code> generates an XML report that contains the details of the <code>Person_Name</code> purpose.</p>
ITEM_DATA	<p>Specifies the sample data for which you want to identify the population file to use. Ensure that you set the item type to <code>suggest_population</code>. Use the sample data that the Unicode format supports.</p> <p>Use the following format:</p> <pre>ITEM_DATA=utf-8 utf-16,&lt;Length&gt;,&lt;Sample Data&gt;</pre> <p>For example, <code>ITEM_DATA=utf-8,15,abc def ghi jkl</code> indicates that the sample data is <code>abc def ghi jkl</code> and the length of sample data is 15 bytes.</p>
MAP_DATA	<p>Specifies the sample data for which you want to identify the SSA-NAME3 field to use. Ensure that you set the item type to <code>map_data</code>. Use the sample data that the Unicode format supports.</p> <p>Use the following format:</p> <pre>MAP_DATA=utf-8,&lt;Repeat Count&gt;,&lt;Length 1&gt;,&lt;Sample Data 1&gt;[,&lt;Length 2&gt;,&lt;Sample Data 2&gt;...,&lt;Length N&gt;,&lt;Sample Data N&gt;]</pre> <p>For example, <code>MAP_DATA=utf-8,2,John Smith,10,John Smithe,11</code> indicates that you use two sets of sample data, one is John Smith and another is John Smithe.</p>
DEBUG_ENABLE=[Y N]	<p>Verifies the input parameters and writes debug information about the parameters to a specified log file.</p> <p>Default is N.</p>
DEBUG_FILE	<p>When <code>DEBUG_ENABLE</code> is set to Y, <code>DEBUG_FILE</code> specifies the name of the log file to which you want to write the debug information about the input parameters.</p>



# CHAPTER 6

## Advanced Controls

This chapter includes the following topics:

- [UNICODE\\_ENCODING, 313](#)
- [PURPOSE, 315](#)
- [SSA-NAME3 Error Messages, 320](#)

### UNICODE\_ENCODING

The **ssan3\_get\_keys\_encoded**, **ssan3\_get\_ranges\_encoded** and **ssan3\_match\_encoded** calls have a Field Data Type parameter that can be used to specify the encoding type. This control is only for use with the **ssan3\_get\_keys**, **ssan3\_get\_ranges** and **ssan3\_match** function calls. The section on UTF-8 considerations is relevant to all API function calls.

This Control is used in the **ssan3\_get\_keys**, **ssan3\_get\_ranges** and **ssan3\_match** function calls, and instructs SSA-NAME3 to accept Unicode data input.

Possible values are:

Encoding	Meaning
TEXT	Data is not in a Unicode encoding
UTF-8	Unicode UTF-8 format
UTF-16	Unicode UTF-16 format
UTF-16LE	Unicode UTF-16 format Little Endian
UTF-16BE	Unicode UTF-16 format Big Endian
UTF-32	Unicode UTF-32 format
UCS-2	Same as Unicode UTF-16 format
UCS-4	Unicode UTF-32 format
CP932	Japanese CP932 Codepage (shift-JIS)
CP936	Chinese CP936 Codepage (GBK or Simplified Chinese)

Encoding	Meaning
CP949	Korean CP949 Codepage
CP950	Chinese CP950 Codepage (Big5 or Traditional Chinese)
UTF8	All keywords can be specified without the hyphen (-)
DBCShost	Japanese - Host, DBCS

There are short forms for these values

Encoding	Meaning
Y	Unicode UTF-8 format
8	Unicode UTF-8 format
6	Unicode UTF-16 format
L	Unicode UTF-16LE format
B	Unicode UTF-16BE format
4	Unicode UCS-4 or UTF-32 format
J	Japanese CP932 codepage (Shift-JIS)
S	Chinese CP936 codepage (Simplified Chinese)
K	Korean CP949 codepage
T	Chinese CP950 codepage (Traditional Chinese)
D	Japanese - Host, DBCS

All call parameters use single byte except for Key Field Data in the **ssan3\_get\_keys** and **ssan3\_get\_ranges** calls and *Search Data* and *File Data* in the **ssan3\_match** call.

When passing Unicode data, all length and offset values must be number of bytes, not number of characters. UTF8 is a variable length encoding so the number of characters represented will have varying lengths depending on the content of the data.

Scatter / Gather Data Format should be used except for UTF8 encoding in which case the following notes should be considered.

### UTF8 Considerations

If the UNICODE UTF8 encoding is being used then as long as (great) care is taken, all parameters can be treated as Unicode. All return parameters are single character UTF8 except for data from the SSA-NAME3 v2 Info calls which may contain national characters.

For Tagged Data Format the UTF8 encoding can only be used if the delimiter character consists of the single byte UTF8 characters. All the standard (non-accented) characters (A-Z and a-z) and digits (0-9) and many punctuation characters are represented in UTF8 as a single 8 bit character.

In UTF8 if the character code is less than 128 (00 to 7F in hex) then it is a single byte character otherwise it could be a 2 to 6 byte character.

Everything between the delimiter characters will be passed unchanged.

It is also possible to use a multi-byte UTF8 character or characters for the DELIMITER.

## PURPOSE

The `PURPOSE` Control specifies the name of the Matching Purpose to use in the Match call. It takes the following form:

```
PURPOSE=(<expression>)
```

Where `<expression>` is one of the following formats:

```
<expression> := <Purpose_Name>  
<expression> := <Purpose_Name>(<Match_level>)  
<expression> := not <expression>  
<expression> := <expression> or <expression>  
<expression> := <expression> and <expression>  
<expression> := (<expression>)
```

**Note:** The `<expression>` requires brackets () whenever there are embedded spaces, that is when using not, and, or.

The simple form of the Purpose `<expression>`

```
PURPOSE=<Purpose_Name>
```

is also the most commonly used format. For example:`PURPOSE=Address` will cause a match to be done on the supplied Address fields to determine the match purpose "same address".

The form of the `<expression>`:`PURPOSE=<Purpose_Name>(<Match_level>)`

For example:`PURPOSE=Address (Conservative)` is the same as specifying:`PURPOSE=Address`  
`MATCH_LEVEL=Conservative`

However, if both ways of specifying a Match Level are used, the value specified locally associated with each of the purposes will take precedence over the match level specified by the `MATCH_LEVEL` keyword. Only purposes that do not have an individual match level specified will adopt the match level specified by the `MATCH_LEVEL` keyword.

Combining multiple expressions, gives the application designer more flexibility in choosing the method in which match decisions and scores are computed.

### Using Filters

A common need for combining multiple expressions is in the use of exact match "filters" to cause candidate records to be rejected early, usually based on the value of some flag.

There are nine (9) pre-defined Filters that can be used with special reserved names Filter1, Filter2, . . . Filter9. Multiple filters are provided in case the Matching need requires to filter on more than one field. The filter values are normally expected to be flags, usually single-characters; however they can be any length up to 255 characters. The matching done on filters is an exact match, with no editing, which is why they are more suited to codes and flags.

There is another control that is used to return a score of 0% when one or both of the fields used for the Filter are empty or blank.

```
FILTER_OPTIONS=A|B|Y|E|S|F|N
```

A = Search or file must be blank.

B = Both search and file must be blank.

Y = Same as E (Either).

E = Search and file may be blank.

S = Search is blank.

F = File is blank.

N = Both search and file must match.

I = Turns on case insensitive match. You can use I with other filters or as a standalone filter.

The default is to return 100% if both fields are blank.

For example, consider a name file that includes both person and company names. This name file also includes a name type ("P" for person and "C" for company). The business needs to be able to search across both name types in the one search (to avoid missing a match if the name type is wrong), but would also benefit from a search within each name type (a "Person" search and a "Company" search).

For this example, a single unpartitioned SSA-NAME3 Key index would be built. (Partitioning by name type would make searching across name types difficult). Such an index would typically use the `Organization_Name` algorithm as it supports a mixture of personal and organization names.

To support the searching within name type, the matching can use the Filter syntax (a `Filtern` Purpose in combination with the real Match Purpose.)

In this example, to cause a Match function to accept only Typical Matches about "People", using the Purpose "same Organization", the following Controls syntax might be used:

```
PURPOSE=(Filter1 AND Organization(Typical))
```

An example of the corresponding Search Data would be:`*Filter1*P*Organization_Name*<search name>***`

And the File Data would look similar:

```
*Filter1*<Name type flag from file>*
Organization_Name*<name from file>***
```

This will filter out candidates that have a name type of "C".

Of course, this only makes sense if the search application has control over injecting the correct name type into the search record.

The format of the `Filtern` in the Search Data and File Data can one of the following:

- a) `*Filtern*<single value>*`
- (b) `*Filtern*<value1>*Filtern*<value2>*. . .
*Filtern*<valuen>*`
- (c) `*Filtern*List(value1,value2,... valuen)*`

**Note:** (b) and (c) give the same result, that is the ability to evaluate multiple values for the one Filter. The `n` in `valuen` can be a value up to 99.

For example, if the people in our imaginary database could have one of two flags (say "P" or "I"), then the corresponding Search Data appear as:`*Filter1*List(P,I)*Organization_Name*<search name>* **`

. . . or this

```
*Filter1*P*Filter1*I*Organization_Name*<search name>***
```

In another example that uses a `Person_Name` index, filtering may be required on two flags, say on Name Type and on Archive Status (a flag that is set to "A" if the record has been archived).

In this case, to cause a Match function to return only Typical Matches about non-archived people, using the Purpose "same Resident", the following Controls syntax might be used:

```
PURPOSE=(Filter1 AND (NOT Filter2) AND
Resident(Typical))
```

An example of the corresponding Search Data would be:

```
*Filter1*List(P,I)*Filter2*A*Person_Name*
<search name>*Address_Part1*
<search street address>*Address_Part2*
<search locality line>***
```

And the File Data would appear as:

```
*Filter1*<Name type flag from file>*Filter2*
<Archive Status flag from file>*Person_Name*
<name from file>*Address_Part1*
<street address from file>*Address_Part2*
<locality line from file>***
```

The List and repeating field formats may be used for both Search and File Data. If this is the case, the matching done will be many-to-many (i.e. not positional).

The Filter construct is particularly useful for IIR systems and DCE projects, where the user does not have control over the `SELECT` statement in candidate selection. For simple SSA-NAME3 Applications, the programmer could achieve the same effect by hard-coding a `WHERE` into the `SELECT` clause. However, for cases where dynamic lists are used for filtering, it may be easier for the SSA-NAME3 programmer to string these into the `PURPOSE` Control of the `ssan3_match` call.

When running Clustering projects, remember that there is not a concept of search and file record, i.e. the clustering process is about grouping file records. Therefore, using the Filter construct in clustering is restricted to using whatever flags are in the file data. This is useful for adding some additional requirements for a record to join a cluster. For example, in the first example in this section, if the mixed person and company file were to be clustered, using filtering would make it possible to ensure only records with the same name type were grouped together.

## [FILTER\\_SEARCHVAL Control](#)

You can use the `FILTER_SEARCHVAL` control to specify a fixed value for the `FilterN` field. When you use `FILTER_SEARCHVAL=value`, the value must match both the search and file records.

The following example matches as the `FILTER1` values for the search and file record are the same and `FILTER_SEARCHVAL` is not specified:

```
Controls: 'PURPOSE=(FILTER1 AND PERSON_NAME) '
Search data: '*filter1*A*Person Name*Mike Taylor***'
File data: '*filter1*A*Person_Name*Mike Taylor***'
Score:
'100'
Decision: 'A'
```

The following example does not match as the `FILTER1` fields do not match the value specified by the `FILTER_SEARCHVAL` control:

```
Controls: 'PURPOSE=(FILTER1 AND PERSON_NAME) FILTER_SEARCHVAL=B'
Search data: '*filter1*A*Person Name*Mike Taylor***'
File data: '*filter1*A*Person_Name*Mike Taylor***'
Score:
'000'
Decision: 'R'
```

The following example matches as both `FILTER1` fields are the same and they match the value specified by the `FILTER_SEARCHVAL` control:

```
Controls: 'PURPOSE=(FILTER1 AND PERSON_NAME) FILTER_SEARCHVAL=B'
Search data: '*filter1*B*Person_Name*Mike Taylor***'
```

```
File data: '*filter1*B*Person_Name*Mike Taylor***'  
Score:  
'100'  
Decision: 'A'
```

## Multi-Purpose Matching

Another use for combining multiple expressions will be to achieve Multi-Purpose Matching.

When multiple Purposes are used, it is important to note the following:

The Multi-Purpose expression is evaluated in a strict left to right order;

Early exit from the match process is only possible after evaluation of the first Purpose;

All Purposes in the expression share the same data as passed in the Search Data and File Data fields.

If two Purposes share the same field and both Purposes need to be evaluated, that field will be evaluated twice (as it may be defined with different match options depending on which Purpose it is in).

One example of Multi-Purpose matching is to create an early exit condition that is likely to improve performance. This would be either an early "accept" where Purposes are joined by an OR, or an early "reject" in the case of an AND. Again, note that early exit from the match process is only possible after evaluation of the first Purpose.

For example, if in a typical Resident purpose it is logically possible to reject on the basis of "Address" not passing a Conservative match, the following might be used:  
`PURPOSE=(Address(Conservative) AND Resident(Typical))`

If the Address purpose receives a "Reject" decision from the Conservative match, the Resident Purpose is not evaluated as the AND has failed. However, if the Address purpose does not result in a Reject condition and the Resident purpose is thus evaluated, note that the address data will be re-matched as part of the Resident purpose, in addition to the `Person_Name` field.

In the above example overall performance (i.e. the run-time of a batch job, or the overall performance of an online system) improves the more often early-exit occurs. Conversely, performance can decrease the more often both Purposes need to be evaluated (because the address field must be evaluated for each Purpose).

Another example of Multi-Purpose matching is to return a super-set of matches, such as:  
`PURPOSE=(Individual OR Resident)`

This will cause matches to be accepted where either the same Individual (Name + (Date of Birth or ID Number)) OR the same Resident (Name + Address) is present. In this example, the `Match_Level` Control will be used to apply to both Purposes.

A third example is for mixing Match Levels. For example:

```
PURPOSE=(Contact(Typical) OR  
Wide_Contact(Conservative))
```

This will accept matches if either `Person_Name`, `Organization_Name` and `Address` match at the `Typical` level, or `Person_Name` and `Organization_Name` match at the `Conservative` level.

In the above two examples, again note the performance impact of both Purposes being evaluated (i.e. based on the fact that certain fields would be evaluated twice).

When combining Purposes with `<and, or, not>`, the Purposes are evaluated in a left-to-right order. If the score/decision from the first Purpose invalidates the expression, then no further processing is done and the process returns with that Purpose's score/decision.

If all Purposes are evaluated, the score/decisions are combined in the following manner:

<expression a> and <expression b>

<b>If &lt;expression a&gt; decision =</b>	<b>and &lt;expression b&gt; decision =</b>	<b>Final decision / Score</b>
Reject	N/A	Reject / <exp. a> Score
Undecided	Accept	Undecided / Score=lowest
Undecided	Undecided	Undecided / Score=lowest
Undecided	Reject	Reject / Score=lowest
Accept	Accept	Accept / Score=lowest
Accept	Undecided	Undecided / Score=lowest
Accept	Reject	Reject / Score=lowest

<expression a> or <expression b>

<b>If &lt;expression a&gt; decision =</b>	<b>or &lt;expression b&gt; decision =</b>	<b>Final decision / Score</b>
Accept	N/A	Accept / <exp. a> Score
Undecided	Accept	Accept / Score=highest
Undecided	Undecided	Undecided / Score= highest
Undecided	Reject	Undecided / Score= highest
Reject	Accept	Accept / Score=highest
Reject	Undecided	Undecided / Score= highest
Reject	Reject	Reject / Score= highest

not <expression>

<b>If &lt;expression&gt; decision not =</b>	<b>Decision / Score returned</b>
Accept	Reject / Score=100-Score
Undecided	Undecided / Score=100-Score
Reject	Accept / Score=100-Score

# SSA-NAME3 Error Messages

The following error messages may be received from an SSA-NAME3 Call. Contact Informatica Corporation if you require assistance resolving the problem.

The following messages refer to problems with the parameters or the formatting of the parameters. You should check that the parameters are correct. Use the message as an indicator as to which parameter to check. Some of the messages refer to a `CONTROLS` keyword which has an invalid or incorrect value.

```
Invalid population parameter
Invalid codepage parameter
Invalid field parameter
Invalid key_level parameter
Invalid offset/length parameter
Length parameter exceeds maximum
Invalid search type parameter
Invalid purpose parameter
Invalid level parameter
Invalid search data layout
Invalid file data layout
Search record does not contain all fields required for Purpose
File record does not contain all fields required for Purpose
Invalid system parameter
Invalid Controls parameter
Field delimiter not specified or is invalid
Invalid key field data layout
Invalid INFO type
Unrecognized keyword found in controls
No keywords found in controls
Keyword not allowed for this call type
Invalid NAMEFORMAT value - must be L or R
Invalid CHARACTER ENCODING value
Key Data Field does not contain any input data
MAX_ENTRIES value is greater than supported size
Invalid KEYSIZE value - must be 5 or 8
Missing field parameter
Missing matching bracket
Syntax error in purpose expression
Syntax error in weight adjustment
Syntax error in weight adjustment, invalid number
Syntax error in weight adjustment, missing fieldname
Invalid value for WSTACK (valid values are B,S or F)
Invalid EARLYEXIT value - must be Y or N
Search record is empty
File record is empty
Input record is empty
level name too long
Invalid SCACHE value - must be Y or N
No populations found in the SSAPR/system directory
No systems found in the SSAPR directory
```

These error messages refer to internal errors and should be referred to Informatica Corporation.

```
Internal error assembling record for matching
Error reported by core
Work area is too small
Not enough memory to allocate workarea
Unable to establish session
Error reading SSAPR directory
Cannot sort in the available memory
Unsupported call - old CALL interface no longer supported
Unable to allocate space for layout
NAMESET word stack overflow
```



These error messages indicate that the number of values that would have been returned was larger than the field size specified to return the values. For example the keys and search ranges have a maximum value of 1024 entries. The Info records maximum size is 99.

```
NAMESET key stack overflow
NAMESET search table overflow
Ranges table overflow
Key table overflow
Info table overflow
```

The following messages refer to problems with the Data Encoding Type. The `UNICODE_ENCODING` control (if used) must match the Data Encoding Type. The specification of the Data Field (using tagged or scatter gather format) must not contain a length that is longer than the value specified in the Data Field length parameter.

```
Specified user encoding does not match UNICODE_ENCODING control
Invalid user encoding has been specified
Layout specifies field beyond limits of input record
Please verify your search data
Please verify your file data
```

The following message indicates that the value specified in the `SEARCH_LIMIT` control was too small for the number of records expected to be returned by the search. Search too wide - increase `SEARCH_LIMIT`

# CHAPTER 7

## Composite Keys

This chapter includes the following topics:

- [Combining multiple fields into one key, 322](#)
- [Key Sizes, 323](#)
- [Limitations, 323](#)
- [Performance, 323](#)
- [Extended Functionality for Composite Key, 324](#)
- [Useful Key and Range Keywords for Composite Keys, 325](#)
- [SSA-NAME3 Error Messages, 327](#)

### Combining multiple fields into one key

Build keys based on more than one SSA-NAME3 field for example Person\_Name plus Code, Person\_Name plus Address\_Part1. Up to five fields may be combined into one composite key.

This produces a more selective key that speeds up searches at the cost of a higher number of keys.

```
FIELD=Field1,Field2 [...,Field5]
```

The input record can be specified using scatter gather or delimited format. To specify using the scatter gather format use the `LAYOUT= control`.

For example:

```
ssan3_get_keys
testpops
ausv102
FIELD=Person_Name,Address_Part1 Key_Level=standard,limited -
LAYOUT=Person_Name,0,10,Person_Name,11,13,Address_Part1,25,10
John Smith James Madison10 Rudd st
```

# Key Sizes

The key sizes are fixed to the default none binary size,8 bytes or 16 bytes.

To get the key size use the following SSA-NAME3 Info call support to check for the key size:

```
ITEM=field.options SELECT=fieldname1,fieldname2[... ,fieldname5]
```

For example:

```
ITEM=field.options SELECT=Person_Name,Address_Part1
```

The length will be returned in the Info data parameter.

```
-----ssa_info-----  
Session Id: 1048576  
System: 'efault'  
Population:'ausv102'  
Controls: 'ITEM=field.options SELECT=Person_Name,Address_Part1 ' Rsp_code:'0'  
SSA_msg: ''  
Count: 1  
0 'keylen=16'
```

# Limitations

Up to 5 fields may be specified for a composite key.

The fields specified in the FIELD= control must be supplied when calling `ssan3_get_keys` or `ssan3_get_ranges`.

Only key sizes of 8 or 16 are allowed (for all standard populations). Key size is 8 for fields such as `Person_Name`, `Address_Part1`, `Organization_Name` etc, and 16 for fields such as `Code`, `Telephone_Number` etc.

To use the new composite key for the build keys and ranges API calls you must use the same FIELD= definition (for example `FIELD=Peson_Name,Address_Part1`).

# Performance

Preliminary benchmarks for two fields show an increase in search performance of 6 to 8 times. The number of records retrieved decreased by a similar amount.

The number of keys can be reduced using `KEY_LEVELS` for example `KEY_LEVEL=standard,limited`.

As an example consider the case where the index was created using `Organization_Name` and the match uses `Organization_Name` and `Address_Part1`.

The composite key could use `Organization_Name,Address_Part1` with `KEY_LEVEL=standard,limited` and then the search could use `KEY_LEVEL=standard SEARCH_LEVEL=Narrow`. This will have a tighter search on

Address\_Part1 which will be similar to a Conservative match when the key was built only on Organization\_Name.

An alternative would be to build the key for Address\_Part1,Organization\_Name with KEY\_LEVEL=limited,standard.

The fewer keys built by the left most fields in the composite keys the smaller the overall number of keys will be. The performance improvement comes from the smaller candidate sets, as the keys will be more selective and less records will need matching. The same (or similar) number of records will be found using composite key with an increase in performance.

Adding extra key / search options using keywords described below will find variations that might be missed (such as for fields with null or empty values - INSSERTNULLCLOSINGRANGE and INSSERTNULLCLOSINGKEY - or variations with initials -PROBESMAJORINIT1). Adding these extra key words at search time will retrieve more candidates with minimal impact on performance overall. See examples below in "keywords" section.

## Extended Functionality for Composite Key

New keywords were added specifically for use with composite key.

### Keywords

The following keywords were added specifically for use with composite key.

INSSERTNULLCLOSINGRANGE and INSSERTNULLCLOSINGKEY

Added to allow a composite key to find records with NULL values.

```
----- ssa_get_ranges-----  
System: 'default'  
Population: 'usa'  
Controls: 'FIELD=Person_Name,Address_Part1 NM3KEYWORDS=Person_Name:INSSERTNULLCLOSINGKEY'  
Key Field Data: '*Person_Name*BELLA ROSE BINDOO ELLIOT*Address_Part1*7993 Ligula***'  
...  
17 'K$$$$$$TAF$?TWO' 'K$$$$$$TAF$?TWR'
```

An additional range is added to search for the case where the person name is missing.

Additional keywords that add additional variations can be found in the section "Useful Key and Range keywords for Composite key" below or in the SSA-NAME3 Customization and Definition guide.

### Updated Controls

The following controls are used with composite keys.

#### FIELD=

To specify more than one field provide a list to the FIELD=control.

```
FIELD=Person_Name,Address_Part1
```

#### **LAYOUT=**

Field name is allowed in LAYOUT control.

```
LAYOUT=Person_Name,0,50,Address_Part1,50,60
```

#### **KEY\_LEVEL=**

Multiple key levels may be specified.

```
KEY_LEVEL=standard,limited
```

KEY\_LEVEL may now be specified when calling `ssan3_get_ranges` for all fields except the rightmost field.

For example, for a `ssan3_get_ranges` call:

```
FIELD=Person_Name,Address_Part1 KEY_LEVEL=limited SEARCH_LEVEL=narrow
```

```
FIELD=Person_Name,Address_Part1,Telephone_Number KEY_LEVEL=limited,standard  
SEARCH_LEVEL=narrow
```

For example, for a `ssan3_get_keys` call (also showing LAYOUT example):

```
FIELD=Person_Name,Address_Part1 Key_Level=standard,limited -
```

```
LAYOUT=Person_Name,0,10,Person_Name,11,13,Address_Part1,25,10
```

#### **KEY\_SIZE=**

A list of key size can be specified.

```
KEY_SIZE=8,8
```

For composite field the code keys (size of 12) will default to 16 and name keys (size of 5, default is 8) will default to 8.

#### **NONULLKEY=Y**

Supports composite key as well as single field key.

#### **UNICODE\_ENCODING**

The field name can now be specified.

```
UNICODE_ENCODING=Person_name:utf- 8'
```

#### **NM3KEYWORDS**

The field name must be specified with composite key otherwise this will be ignored.

```
NM3KEYWORDS=Person_Name:INSERTNULLCLOSINGKEY is valid.
```

However, `NM3KEYWORDS=INSERTNULLCLOSINGKEY` will be silently ignored.

## Useful Key and Range Keywords for Composite Keys

The key and range keywords for composite keys are listed below.

#### **INSERTNULLCLOSINGRANGE**

Return a bad range closing entry.

#### **INSERTNULLCLOSINGKEY**

Return an extrabad key closing entry.

### **PROBESMAJORINITALL**

Enables PROBESMAJORINIT1, PROBESMAJORINIT2 and PROBESMAJORINIT3.

### **PROBESMAJORINIT1**

Build probes (or keys) changing Minors to initials.

Builds extra probes or keys by changing each minor to an initial and building Initial Plus Major pairs for example JOHN ANDREW SMITH, builds probes for J SMITH and A SMITH.

### **PROBESMAJORINIT2**

Build probes (or keys) changing Minors to initials.

Builds an extra probe or key after changing all minors to an initial for example JOHN ANDREW SMITH builds probe for J A SMITH.

### **PROBESMAJORINIT3**

Build probes (or keys) changing Minors to initials.

Builds extra probes or keys after changing one minor at a time to an initial for example JOHN ANDREW SMITH builds probes for J ANDREW SMITH and JOHN A SMITH.

### **EXTRAWORDPROBE**

Build an extra key or probe without the right-most word when the name has at least three words.

### **PROBESWORD**

Add probes for each word.

For example with the name JOHN ANDREW SMITH three probes will be generated in addition to the normal Search-table, one for JOHN, one for ANDREW and another for SMITH.

Applies to Person\_Name, Address\_Part1, Company\_Name, Organization\_Name (not KEY\_LEVEL=Limited, all SEARCH\_LEVELS), Product\_Name, Product\_Description (all KEY\_LEVELS and SEARCH\_LEVELS)

### **PROBESINIT**

Add probes for each word + initial.

Using JOHN ANDREW SMITH, probes would be generated for the following word/initial pairs: ANDREW J, JOHN S, SMITH A, SMITH J, ANDREW S, JOHN A.

### **PROBESALL**

Shorthand method of specifying both PROBESWORD and PROBESINIT.

### **PROBESMAJ**

Generate a probe for the major word only.

**Note:** PROBES\* key words do not apply to Code fields like VIN, ISBN10, ISBN13, Telephone\_Number, Credit\_Card, Code or to Geocode and Date.

## **Example**

We have a search where we want to find records with the same Telephone Number or records with no Telephone Number. If we have a composite key that includes the Telephone Number then it will not find the record where the number is missing unless we add a range.

When building the index we can use:

```
FIELD=Person_Name,Telephone_Number
```

This will build composite keys using two fields, `Person_Name` and `Telephone_Number`.

At search time we add the keyword to build the NULL range for `Telephone_Number` so that a range will be built with a NULL value.

```
FIELD=Person_Name,Telephone_Number NM3KEYWORDS=Telephone_Number:INSERTNULLCLOSINGRANGE
```

Without the `NM3KEYWORDS` control and `INSERTNULLCLOSINGRANGE` records with no Telephone Number will not be found.

**Note:** The `Person_Name` part of the composite key will not be the NULL key - that can be added using `INSERTNULLCLOSINGKEY`. To avoid the range where all parts of the composite key are the NULL key you can use the `NONULLKEY=Y` control.

## SSA-NAME3 Error Messages

The following error messages may be received from an SSA-NAME3 Call. Contact Informatica LLC if you require assistance resolving the problem.

The following messages refer to problems with the parameters or the formatting of the parameters. You should check that the parameters are correct. Use the message as an indicator as to which parameter to check. Some of the messages refer to a `CONTROLS` keyword which has an invalid or incorrect value.

Invalid population parameter

Invalid codepage parameter

Invalid field parameter

Invalid key\_level parameter

Invalid offset/length parameter

Length parameter exceeds maximum

Invalid search type parameter

Invalid purpose parameter

Invalid level parameter

Invalid search data layout

Invalid file data layout

Search record does not contain all fields required for Purpose

File record does not contain all fields required for Purpose

Invalid system parameter

Invalid Controls parameter

Field delimiter not specified or is invalid

Invalid key field data layout

Invalid INFO type

Unrecognized keyword found in controls

No keywords found in controls

Keyword not allowed for this call type  
Invalid NAMEFORMAT value - must be L or R  
Invalid CHARACTER ENCODING value  
Key Data Field does not contain any input data  
MAX\_ENTRIES value is greater than supported size  
Invalid KEYSIZE value - must be 5 or 8  
Missing field parameter  
Missing matching bracket  
Syntax error in purpose expression  
Syntax error in weight adjustment  
Syntax error in weight adjustment, invalid number  
Syntax error in weight adjustment, missing fieldname  
Invalid value for WSTACK (valid values are B,S or F)  
Invalid EARLYEXIT value - must be Y or N

Search record is empty  
File record is empty  
Input record is empty  
level name too long  
Invalid SCACHE value - must be Y or N  
No populations found in the SSAPR/system directory  
No systems found in the SSAPR directory

**These error messages refer to internal errors and should be referred to Informatica LLC.**

Internal error assembling record for matching  
Error reported by core  
Work area is too small  
Not enough memory to allocate workarea  
Unable to establish session  
Error reading SSAPR directory  
Cannot sort in the available memory  
Unsupported call - old CALL interface no longer supported  
Unable to allocate space for layout  
NAMESET word stack overflow

**These error messages indicate that the number of values that would have been returned was larger than the field size specified to return the values. For example the keys and search ranges have a maximum value of 1024 entries. The Info records maximum size is 99.**

NAMESET key stack overflow  
NAMESET search table overflow



Ranges table overflow

Key table overflow

Info table overflow

The following messages refer to problems with the Data Encoding Type. The `UNICODE_ENCODING` control (if used) must match the Data Encoding Type. The specification of the Data Field (using tagged or scatter gather format) must not contain a length that is longer than the value specified in the Data Field length parameter.

Specified user encoding does not match `UNICODE_ENCODING` control

Invalid user encoding has been specified

Layout specifies field beyond limits of input record

Please verify your search data

Please verify your file data

The Following message indicates that the value specified in the `SEARCH_LIMIT` control was too small for the number of records expected to be returned by the search.

Search too wide - increase `SEARCH_LIMIT`

## CHAPTER 8

# Address Standardization

The Address Standardization APIs are used to parse addresses into their components fields and to validate them against postal reference databases. They can only be used successfully if the separately licensable *Address Standardization Module* has been installed.

This section of the documentation describes the functions, semantics, process flow, parameter values / setting and returned data.

## Initialization

A session must first be created by calling **ssan3\_open**.

Each caller must initialize the Address Standardization sub-system by calling **ssan3\_addr\_init**. This API must be called prior to calling any other Address Standardization APIs.

**Note:** By default ASM uses AddressDoctor v4. To use ASM with AddressDoctor v5 call **ssan3\_addr\_set\_option** API prior to calling **ssan3\_addr\_init**. Refer below table for **ssan3\_addr\_set\_option**:

Option Name	Option Value	Meaning
LIBRARY_VERSION	V5	Use ASM with AddressDoctor v5
	V4	Use ASM with AddressDoctor v4

## Character Sets and Countries

Initialization is usually followed by a call to **ssan3\_addr\_set\_attrib**. This is used to define the character set of the input data. The parsed / validated address fields will be returned to the caller using this character set as well. A list of supported character sets appear below for ASM using AddressDoctor v4:

Character Set
UTF32BE
UTF32LE

Character Set
UTF16BE
UTF16LE
UTF8
UCS4BE
UCS4LE
UCS2BE
UCS2LE
ISO8859_1
ISO8859_2
ISO8859_3
ISO8859_4
ISO8859_5
ISO8859_6
ISO8859_7
ISO8859_8
ISO8859_9
ISO8859_10
ISO8859_15
IBM437
IBM850
IBM852
IBM855
IBM857
IBM862
WIN1250
WIN1251
WIN1252

Character Set
WIN1253
WIN1254
WIN1255
WIN1256
WIN1257
BIG5
JIS
GBK
UHC
ASCII
EBCDIC

**Note:** Character sets like UTF32BE, UTF32LE, UCS4BE and UCS4LE not supported by ASM using AddressDoctor v5.

As some of these character sets are multi-byte, **ssan3\_addr\_set\_lines** and **ssan3\_addr\_get\_field\_idx** transfer address data using the Block data type (fixed length, non-nul terminated pieces of memory).

The **ssan3\_addr\_set\_attrib** API is also used to specify a default country. This value is used only when address parsing fails to find a valid country within the address. A list of valid country names appear below:

Country Name
Aruba
Afghanistan
Angola
Anguilla
Albania
Andorra
Netherlands Antilles
United Arab Emirates
Argentina
Armenia

<b>Country Name</b>
American Samoa
Antarctica
French Southern Territories
Antigua and Barbuda
Australia
Austria
Azerbaijan
Burundi
Belgium
Benin
Burkina Faso
Bangladesh
Bulgaria
Bahrain
Bahamas
Bosnia and Herzegovina
Belarus
Belize
Bermuda
Bolivia
Brazil
Barbados
Brunei Darussalam
Bhutan
Bouvet Island
Botswana
Central African Republic

Country Name
Canada
Cocos Islands
Switzerland
Chile
China
Cte d'Ivoire
Cameroon
Congo, The Democratic Republic of the
Congo
Cook Islands
Colombia
Comoros
Cape Verde
Costa Rica
Cuba
Christmas Island
Cayman Islands
Cyprus
Czech Republic
Germany
Djibouti
Dominica
Denmark
Dominican Republic
Algeria
Ecuador
Egypt

Country Name
Eritrea
Western Sahara
Spain
Estonia
Ethiopia
Finland
Fiji
Falkland Islands
France
Faroe Islands
Micronesia, Federated States of
Gabon
United Kingdom
Georgia
Ghana
Gibraltar
Guinea
Guadeloupe
Gambia
GuineaBissau
Equatorial Guinea
Greece
Grenada
Greenland
Guatemala
French Guiana
Guam

<b>Country Name</b>
Guyana
Hong Kong
Heard Island and McDonald Islands
Honduras
Croatia
Haiti
Hungary
Indonesia
India
British Indian Ocean Territory
Ireland
Iran, Islamic Republic of
Iraq
Iceland
Israel
Italy
Jamaica
Jordan
Japan
Kazakstan
Kenya
Kyrgyzstan
Cambodia
Kiribati
Saint Kitts and Nevis
Korea, Republic of
Kuwait



<b>Country Name</b>
Lao People's Democratic Republic
Lebanon
Liberia
Libyan Arab Jamahiriya
Saint Lucia
Liechtenstein
Sri Lanka
Lesotho
Lithuania
Luxembourg
Latvia
Macau
Morocco
Monaco
Moldova, Republic of
Madagascar
Maldives
Mexico
Marshall Islands
Macedonia, The Former Yugoslav Republic of
Mali
Malta
Myanmar
Mongolia
Northern Mariana Islands
Mozambique
Mauritania

<b>Country Name</b>
Montserrat
Martinique
Mauritius
Malawi
Malaysia
Mayotte
Namibia
New Caledonia
Niger
Norfolk Island
Nigeria
Nicaragua
Niue
Netherlands
Norway
Nepal
Nauru
New Zealand
Oman
Pakistan
Panama
Pitcairn
Peru
Philippines
Palau
Papua New Guinea
Poland

Country Name
Puerto Rico
Korea, Democratic People's Republic of
Portugal
Paraguay
Palestinian Territory, Occupied
French Polynesia
Qatar
Runion
Romania
Russian Federation
Rwanda
Saudi Arabia
Sudan
Senegal
Singapore
South Georgia and the South Sandwich Islands
Saint Helena
Svalbard and Jan Mayen
Solomon Islands
Sierra Leone
El Salvador
San Marino
Somalia
Saint Pierre and Miquelon
So Tom and Prncipe
Suriname
Slovakia

<b>Country Name</b>
Slovenia
Sweden
Swaziland
Seychelles
Syrian Arab Republic
Turks and Caicos Islands
Chad
Togo
Thailand
Tajikistan
Tokelau
Turkmenistan
Timor-Leste
Tonga
Trinidad and Tobago
Tunisia
Turkey
Tuvalu
Taiwan, Province of China
Tanzania, United Republic of
Uganda
Ukraine
United States Minor Outlying Islands
Uruguay
United States
Uzbekistan
Holy See

Country Name
Saint Vincent and the Grenadines
Venezuela
Virgin Islands, British
Virgin Islands, U.S.
Viet Nam
Vanuatu
Wallis and Futuna
Samoa
Yemen
Serbia and Montenegro
South Africa
Zambia
Zimbabwe

The character set and default country are used for the life of the session or until they are changed.

## Providing an Input Address

Address data may be provided in "10 line format", by calling **ssan3\_addr\_set\_lines**. If "5 line format" similar to its appearance on an envelope is to be used, the remaining lines should be passed as empty string.

When calling the CASS certified engine (`Val_Mode=Certify`), only 4 lines of input are accepted. Any additional lines will be ignored. The input lines must correspond to

- Organization,
- Delivery Address,
- Locality (City, State and Zip), and
- Country

## Parsing an Address

Parsing is the process of separating the address lines into separate address fields. It does not check whether or not the address fields constitute a valid postal address.

A call to `ssan3_addr_parse` returns a Long Array of field lengths. These correspond to the lengths of the parsed address fields, as per the table below. Fields with a length of zero simply mean that a particular address component was not present (for example, a middle name).

Field Index	Address Field
0	Organization
1	Department
2	Nobility (e.g. Lord)
3	Title (e.g. Mr)
4	First Name
5	Middle Name
6	Last Name
7	Function (e.g. Manager)
8	Building
9	Sub-Building
10	Street_1
11	Street_2
12	House Number
13	P.O. Box
14	Dependent Locality (e.g. URB, Colonia)
15	Locality (e.g. County)
16	Province (e.g. State)
17	Zip Code
18	Country
19	Double Dependent Locality
20	Sorting Code (Mail Sort)
21	County Information
22	Zip+4
23	Geocoding Latitude
24	Geocoding Longitude

Field Index	Address Field
25	Geocoding Unit
26	Residue

**Note:** Some fields will not return values for ASM with AddressDoctor v4, since they may not be supported (i.e. geocoding related fields).

For example, field length [13] represents the length of the "PO Box" component of the address.

## Validating an Address

Validating involves parsing the address into its component fields and checking those fields against a postal validation database. `ssan3_addr_validate` returns a `status` that indicates whether or not the address was valid, and if not, whether it could be corrected and its likelihood of being delivered successfully. Refer the table below for validation status for ASM using AddressDoctor v4:

Status Code	Meaning
0	Address is correct
1	Address was corrected
2	Needs correction; deliverability high
3	Needs correction; deliverability fair
4	Needs correction; deliverability small
5	Country not recognized
6	No valid country database found
7	Country not unlocked
8	No validate called yet
9	Insufficient information
10	No suggestions
11	Suggestions incomplete
12	Suggestions

Refer below table for validation status for ASM using AddressDoctor v5:

Status Code	Meaning
0	Verified - Input data correct - all elements were checked and input matched perfectly
1	Verified - Input data correct on input but some or all elements were standardised or input contains outdated names or exonyms
2	Verified - Input data correct but some elements could not be verified because of incomplete reference data
3	Verified - Input data correct but the user standardisation has deteriorated deliverability
4	Corrected - all elements have been checked
5	Corrected - but some elements could not be checked
6	Corrected - but delivery status unclear
7	Corrected - but delivery status unclear because user standardisation was wrong
8	Data could not be corrected completely, but is very likely to be deliverable - single match
9	Data could not be corrected completely, but is very likely to be deliverable - multiple matches
10	Data could not be corrected, but there is a slim chance that the address is deliverable
11	Data could not be corrected and is pretty unlikely to be delivered
12	FastCompletion Status - Suggestions are available - complete address
13	FastCompletion Status - Suggested address is complete but combined with elements from the input
14	FastCompletion Status - Suggested address is not complete
15	FastCompletion Status - Insufficient information provided to generate suggestions
16	Country recognized from ForceCountryISO3 Setting
17	Country recognized from DefaultCountryISO3 Setting
18	Country recognized from name without errors
19	Country recognized from name with errors
20	Country recognized from territory
21	Country recognized from province
22	Country recognized from major town
23	Country recognized from format
24	Country recognized from script
25	Country not recognized - multiple matches



Status Code	Meaning
26	Country not recognized
27	Parsed perfectly
28	Parsed with multiple results
29	Parsed with Errors - Elements change position
30	Parse Error - Input Format Mismatch
31	Validation Error: No validation performed because country was not recognized
32	Validation Error: No validation performed because required reference database is not available
33	Validation Error: No validation performed because country could not be unlocked
34	Validation Error: No validation performed because reference database is corrupt or in wrong format
35	Validation Error: No validation performed because reference database is too old - contact AddressDoctor to obtain updated reference data

The validation process may generate a number of suggested addresses when the input address is ambiguous. For example, the address

```
1520 Chestnut
Anytown AT 12345
USA
```

may be ambiguous because both 1520 Chestnut Drive and 1520 Chestnut Court exist and without additional information, we cannot tell them apart. The number of suggestions returned is an output parameter from the validate call.

## Retrieving Address Fields

After parsing or validation, individual address fields are available for collection as part of a "suggestion". Suggestion 0 always holds the parsed fields. Suggestions numbered 1 and above hold the validated address fields.

Individual fields are retrieved one by one using **ssan3\_addr\_get\_field\_idx** by nominating the suggestion index and field index. As some fields may be missing, only those fields that have a non-zero length (as determined by calling **ssan3\_addr\_parse** or **ssan3\_addr\_get\_field\_info**) should be retrieved.

After validation, a field may be retrieved with **ssan3\_addr\_get\_field**. This API is similar to **ssan3\_addr\_get\_field\_idx** but returns two extra codes that describe how the particular field matched the

validation data (`val_status`) and whether or not it was changed by the validation process (`val_mods`). Refer below table for `val_status` and `val_mods` for ASM using AddressDoctor v4:

<code>val_status</code>	Meaning
0	Empty
1	Not found
2	Not Checked (no reference data or no chance of success)
3	Matched with errors
4	Matched without errors

<code>val_mods</code>	Meaning
0	Empty
1	Not checked
2	Not checked but standardized
3	Checked and corrected (changed or inserted)
4	Validated, but changed (synonyms, old names)
5	Validated, but standardized
6	Validated and unchanged

Refer below table for `val_status` and `val_mods` for ASM using AddressDoctor v5:

<code>val_status</code>	Meaning
0	Empty
1	Not found
2	Not checked (no reference data)
3	Wrong - Set by validation only
4	Match with errors in this element

<b>val_status</b>	<b>Meaning</b>
5	Match with changes
6	Match without errors

<b>val_modes</b>	<b>Meaning</b>
0	Empty
1	Not validated and not changed
2	Not validated bus standardized
3	Validated but not changed due to invalid input
4	Validated but not changed due to lack of reference data
5	Validated but not changed due to multiple matches
6	Validated and changed by eliminating the input value
7	Validated and changed due to correction based on reference data
8	Validated and changed by adding value based on reference data
9	Validated, not changed, but delivery status not clear
12	Validated, verified but changed due to outdated name
13	Validated, verified but changed from exonym to official name
14	Validated, verified but changed due to standardization based on casing or language
15	Validated, verified and not changed due to perfect match

## Setting Options

Some optional aspects of Address Standardization behaviour may be set with the **ssan3\_addr\_set\_option** API.

<b>Option Name</b>	<b>Option Value</b>	<b>Meaning</b>
Val_Mode	Correct	Correct the input address. Do not generate suggestions. Generally used in batch mode.
	Suggest	Generate suggestions (the default). Generally used by online applications where the operator can choose between a list of possibilities.

Option Name	Option Value	Meaning
	Complete	Use an incomplete address (fragment) to quickly generate suggestions. Used for online "fast completion" style of applications.
	Certify	Use CASS certified validation rules defined by the USPS. Certify mode is only available for US addresses and requires additional database files to be installed in the DB directory.
Force_Country	True	Force the use of the Default Country
	False	Use Country detected from parsing the Address. This is the Default.
FORMAT_ZIP	True	To format the zip code in country specific format
	False	To unset the format of zip code in country specific format. This is the Default.
BASE_ZIP	True	To retrieve the zip code in Base format
	False	To unset the base format of zip code. This is the Default.

## Getting Options

Some aspects of Address Standardization behavior may be obtained with the **ssan3\_addr\_get\_option** API.

Option Name	Option Value	Meaning
VERSION	N/A	Gets the version number of the validation database

## Sample Code

Sample code that demonstrates the API calls is in the samples directory `ssaas/ad/samples`.

Compilation and linking instructions can be found in the comment block at the beginning of the samples.

To run using the Name3 DLL no parameters are required and it can be run as follows:`addrstd`

To run using the Name3 Server the `host:port` of the Name3 Server must be supplied on the command line. For example,

```
addrstd -h%SSA_N3HOST%
```

The sample code by default makes use of ASM with AddressDoctor v5. To use ASM with AddressDoctor v4 run sample code with `-v4` command line option.

For example for Name3 Server mode:

```
addrstd -h%SSA_N3PORT% -v4
```

For example for Name3 DLL mode:`addrstd -v4`

The sample code makes use of the Swiss validation database. This should be installed prior to use. See the *Validation Database Files* section for details.

## Validation Database Files

The validation process makes use of postal validation databases stored in a sub-directory of SSA-NAME3's installation named `ssaas/ad/ad/db` for ASM using AddressDoctor v4 and `ssaas/ad5/ad/db` for ASM using AddressDoctor v5.

The Address Standardization Module installer does not install any validation databases. They are ordered separately by contacting Informatica Corporation. When you receive these files (named \*.MD), you must copy them into the directory specified above. You will also receive a file named `key` that contains an unlock code for your specific databases. This file must be copied to the same directory.

Also in case of ASM using AddressDoctor v5 to support geocoding, unlock code for geocoding should be present in file named `key.geo`. This file must be copied to the same directory where `key` is copied.

### ASM configuration for AddressDoctor v5

The values of unlock key, number of threads, maximum number of address object to be used for Address Standardization are determined from the values in the file `ssaasmv5.xml`.

**Note:** The format of the ASM ADv5 configuration XML should be in UTF-8.

Address Standardization XML configuration file samples:

The simplest XML configuration file:

```
<?xml version="1.0" encoding="UTF-8"?>
<ASM_Adv5_Config>
  <MAX_THREAD>1</MAX_THREAD>
  <MAX_ADOBJECTS>2</MAX_ADOBJECTS>
  <AD5_UNLOCK_CODE>
  <UNLOCK_CODE>**Placeholder for UnlockCode**</UNLOCK_CODE>
</AD5_UNLOCK_CODE>
</ASM_Adv5_Config>
```

**Note:** `MAX_THREAD` value should not be set to a larger value than the number of cores/CPUs. Its recommended to set the value of `MAX_ADOBJECTS` as twice the number of threads set using `MAX_THREAD`. Also these options should be set before starting Name3 servers.

XML Configuration with multiple license key:

```
<?xml version="1.0" encoding="UTF-8"?>
<ASM_Adv5_Config>
  <MAX_THREAD>1</MAX_THREAD>
  <MAX_ADOBJECTS>2</MAX_ADOBJECTS>
  <AD5_UNLOCK_CODE>
    <UNLOCK_CODE>**Placeholder for UnlockCode 1**</UNLOCK_CODE>
    <UNLOCK_CODE>**Placeholder for UnlockCode 2**</UNLOCK_CODE>
    ...
    <UNLOCK_CODE>**Placeholder for UnlockCode N**</UNLOCK_CODE>
  </AD5_UNLOCK_CODE>
</ASM_Adv5_Config>
```

XML configuration with preload options:

```
<?xml version="1.0" encoding="UTF-8"?>
<ASM_Adv5_Config>
  <MAX_THREAD>1</MAX_THREAD>
```

```

    <MAX_ADOBJECTS>2</MAX_ADOBJECTS>
    <AD5_UNLOCK_CODE>
      <UNLOCK_CODE>**Placeholder for UnlockCode 1**</UNLOCK_CODE>
    </AD5_UNLOCK_CODE>
    <PRELOAD_COUNTRIES>
      <PRELOAD TYPE="FULL">Switzerland</PRELOAD>
      <PRELOAD TYPE="PARTIAL" VALMODE="CORRECT">Canada</PRELOAD>
      <PRELOAD VALMODE="CERTIFY">Australia</PRELOAD>
    </PRELOAD_COUNTRIES>
  </ASM_Adv5_Config>

```

**Note:** Refer to *Character Sets and Countries* section for valid country name for preload.

XML configuration with enrichment options:

```

<?xml version="1.0" encoding="UTF-8"?>
<ASM_Adv5_Config>
  <MAX_THREAD>1</MAX_THREAD>
  <MAX_ADOBJECTS>2</MAX_ADOBJECTS>
  <AD5_UNLOCK_CODE>
    <UNLOCK_CODE>**Placeholder
for UnlockCode 1**</UNLOCK_CODE>
  </AD5_UNLOCK_CODE>
  <PRELOAD_COUNTRIES>
    <PRELOAD
TYPE="PARTIAL" VALMODE="CERTIFY">Australia</PRELOAD>
  </PRELOAD_COUNTRIES>
  <PROCESS_OPTION>
    <ENRICHMENT_OPTION>EnrichmentGeoCoding</ENRICHMENT_OPTION>
    <ENRICHMENT_OPTION>EnrichmentCASS</ENRICHMENT_OPTION>
  </PROCESS_OPTION>
</ASM_Adv5_Config>

```

**Note:** To get CASS, GeoCoding, AMAS, SERP enrichment fields we need to specify enrichment options. EnrichmentGeoCoding, EnrichmentCASS, EnrichmentAMAS, EnrichmentSERP, EnrichmentSNA etc. are the possible values for enrichment option.

## Batch Test Utility

Provided with the Address Standardization Module is the batch utility `asnm3`. This program utilizes all of the API functions. It takes address from an input file and can perform both parsing and validation. It is ideal for verifying programs that utilize the Address Standardization Module API or for batch processing a small number of addresses.

```
%SSABIN%\asnm3 InputFile -hHostName:PortNumber [Options]
where
```

### InputFile

The input file will consist of addresses in a ten line format. Addresses with less than 10 lines must be terminated with a comment line. This is a line beginning with the comment string which defaults to the '#' character.

### -hHostName:PortNumber

NM3 Server host name and port number

Possible options are

-a	Prints the suggested address label.
-A	Force the use of archive tables.

-b	Force the use of ASM with Address Doctor v5 (default v4).
-cCharSet	The Character set to use. The default is WIN1250.
-dDefaultCountry	The Country to use when parsing cannot determine a country from the address.
-f	Force the use of the DefaultCountry.
-gFileName	CASS summary report file-name
-i	Individual field-level input. When specified, each input address consists of n lines, where n is the maximum number of Field-Index values supported, as documented in the <i>Parsing an Address</i> section. Each field value must be prefixed by 3 bytes (which are ignored). This reserves space for a 2 digit fieldindex plus a space, used for documentation purposes

For example, a valid input file appears as follows:

```

00
01
02
03 Mr.
04 John
05 Peter
06 Smith
07
08
09
10 Main Street S
11
12 100 Apt 23A
13
14
15 New York
16 NY
17 10023
18 USA
#

```

Option	Description
-Lpreload_country	The Country to use during preloading of database into memory.
-mValMode	The Mode to use for validation purposes valid values are <i>Suggest</i> , <i>Correct</i> and <i>Complete</i> . The default value is <i>Suggest</i> .
-o	PO Box complete flag.
-pPopulation	The SSA-Name3 Population to use.
-P	Force the use of partially fielded. This should be accompanied with Individual field-level input.
-sSystem	The NM3 System Directory
-S	Prints the address match score.
-rString	Set the comment string. This defaults to #.
-Tpreload_type	Set the preload type. This defaults to <i>NOPRELOAD</i> .

Option	Description
-v	Performs Validation. The default is to parse only.
-xml	Generate CASS 3553 summary report in xml format.
-y	Set preferred language type.
-z	Prints the Validation database version.
-ZzipFormat	Set the format of the output zip code. The valid values are <code>BASE_ZIP</code> and <code>FORMAT_ZIP</code> .



# CHAPTER 9

## ASM Workbench

This chapter includes the following topics:

- [Introduction, 353](#)
- [Launching the ASM Workbench, 353](#)
- [ASM Workbench Input Options, 354](#)
- [ASM Workbench and Batch Test utility, 366](#)

### Introduction

The ASM Developer's Workbench is a Java GUI tool that helps a programmer prototype Address Standardization API calls. The ASM Workbench is used to parse addresses into their individual component fields and to validate them against postal reference databases. The ASM Workbench is used for:

- Parsing Unfielded Address Format
- Validating Fielded and Unfielded Address Format

In order to use the ASM Developer's Workbench, SSA-NAME3 core modules should have been installed, either locally, or on another computer/server.

### Launching the ASM Workbench

ASM Workbench can be launched from SSA-NAME3 %SSABIN% directory.

#### Command line startup for ASM Workbench

To run ASM Workbench from command line you will need to establish the environment by running %SSABIN%\env\iss.bat and start the server. Once your environment is established and servers are up and running, execute `asmcli`. Following are the main input parameters for launching SSA-NAME3 Workbench:

```
asmcli -hHostName:PortNumber -1File1 -2File2 [options]
```

where

**-hHostName:PortNumber**

Search Server Host name and Port Number.

**1File1**

Specifies the file where log message are redirected.

**2File2**

Specifies the file where error message are redirected.

**-b**

Use ASM with AddressDoctor v5.

**-cCharacterSet**

The Character set to use. The default is WIN1250.

**-dDefaultCountry**

The Country to use when parsing can not determine a country from the address.

**-mValidationMode**

The Mode to use for validation purposes valid values are Suggest, Correct, Complete and Certify. The default value is Suggest.

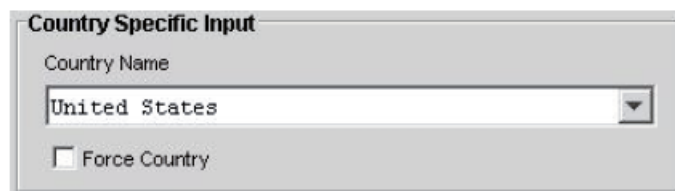
For example:

```
%SSABIN%\asmcli -h%SSA_N3HOST% -1asmcli.log -2asmcli.err -d"United States" -pusa -sdefault
```

## ASM Workbench Input Options

### Country Specific Input

Country specific Input:



The image shows a dialog box titled "Country Specific Input". Inside the dialog, there is a label "Country Name" above a dropdown menu. The dropdown menu currently displays "United States". Below the dropdown menu, there is a checkbox labeled "Force Country", which is currently unchecked.

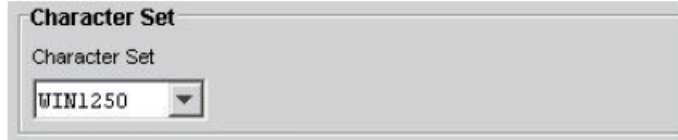
Force Country option

**Force country** option is used to force the use of default country.

**Note:** Before selecting a Country Name for validating an address or preloading country database into memory, appropriate Postal database (.MD) file must be present in %SSATOP%/ssaas/ad/ad/db directory for ASM using AddressDoctor v4 and in %SSATOP%/ssaas/ad5/ad/db directory for ASM using AddressDoctor v5.

## Character Set

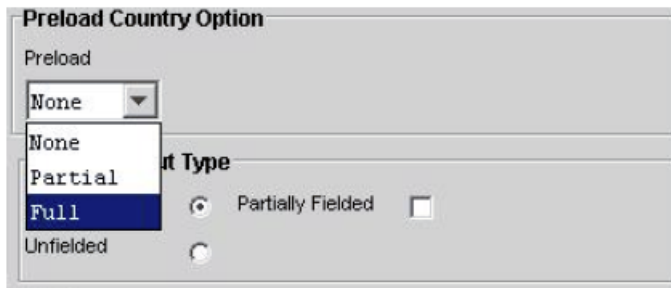
This is used to define the character set of the input data. The parsed / validated address fields will be returned to the caller using this character set as well.



The image shows a dialog box titled "Character Set". It contains a label "Character Set" and a dropdown menu with "WIN1250" selected.

## Country Preload Option

**Preload** option provides greater flexibility in loading the country address database into memory. The preload option includes partial preload for US CASS (certified) including ZIP move and EWS data. Only one country database can be preloaded. If database file is not located or insufficient memory, causes preload to fail.



The image shows a dialog box titled "Preload Country Option". It has a "Preload" dropdown menu with "None" selected. Below it, there are radio buttons for "Fielded", "Partially Fielded", and "Unfielded". The "Fielded" radio button is selected. There is also a checkbox for "Partially Fielded" which is unchecked.

### Partial Preload

Partial preloading will load the data and indexing structures into memory. The reference data itself will remain on the hard drive. Partial is an alternative when not enough memory is available to fully load the desired databases.

### Full Preload

Full preloading will move the entire reference database into memory. This may need a significant amount of memory for countries with large databases such as USA or, but it will increase the processing speed significantly.

**Note:** Before preload of country verify selected country name contains corresponding database (.MD) file located in appropriate postal reference database directory %SSATOP%/ssaas/ad/ad/db for ASM using AddressDoctor v4 and %SSATOP%/ssaas/ad5/ad/db for ASM using AddressDoctor v5.

## Address Input Type



The image shows a dialog box titled "Address Input Type". It has radio buttons for "Fielded", "Partially Fielded", and "Unfielded". The "Unfielded" radio button is selected. There is also a checkbox for "Partially Fielded" which is unchecked.

### Fielded address Input

Fielded addresses will typically provide the most reliable results when cleansing an address. This address input provide separate field for each address component input.

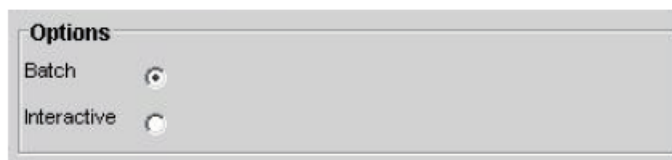
### Partially Fielded Input

In many databases address data has been partially broken out. for example a separate state, or postal code field. But some of the address is left in generic "address lines". In this case the address data are input using the fielded data (example: Contact, Province, Locality, Country, Postal Code) by selecting Fielded address input type, and then use the DeliveryAddressLines to input the ADDRESS\_LINE\_\* elements, this is done by selecting Partially Fielded checkbox.

### UnFielded Input

UnFielded Address Input has no explicit structure (other than 10 line input) this input is most flexible, but produce least reliable results.

## Options



The image shows a dialog box titled "Options". It contains two radio button options: "Batch" and "Interactive". The "Batch" radio button is selected, indicated by a small black dot in the center of the circle.

### Batch Mode

Batch Mode is mainly used for importing input file containing unfielded input data and correcting the input address.

### Interactive Mode

Interactive Mode is user driven, user has to use either unfielded (10 line input) or fielded inputs (individual components address input).

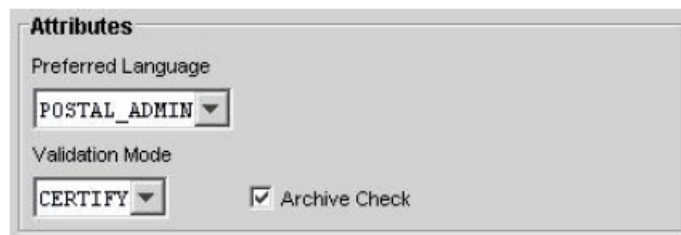
## Parsing and Validation Frame

ASM Workbench provides four different operations for parsing, validating, certify input address and reset for clearing the input fields. For Fielded Address Format type Parse button will be disabled and for certify validation mode Validation button will be disabled.



The image shows a horizontal frame containing four buttons: "Parse", "Validate", "Certify", and "Reset". The "Parse" and "Validate" buttons are disabled, indicated by a light gray color and a faint border. The "Certify" and "Reset" buttons are active, indicated by a darker gray color and a clear border.

## Attributes



The image shows a dialog box titled "Attributes". It contains two dropdown menus and one checkbox. The "Preferred Language" dropdown menu is set to "POSTAL\_ADMIN". The "Validation Mode" dropdown menu is set to "CERTIFY". The "Archive Check" checkbox is checked, indicated by a small black checkmark in the box.

### Preferred Language

This option is used to represent address into appropriate language type.

Option Value	Meaning
POSTAL_ADMIN	Set preferred language to that which is preferred by the postal service. This is the default.
LATIN_SCRIPT	Return the address using Latin script.
ENGLISH	English version of the address.

### Validation Mode

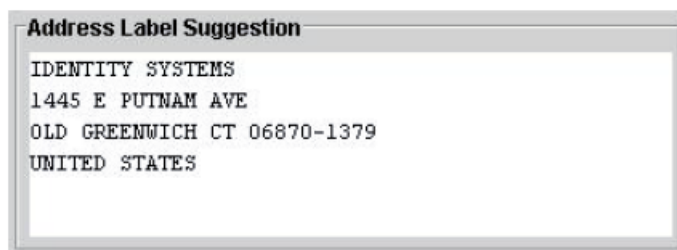
Some optional aspects of Address Standardization behavior may be set by selecting the Validation Mode combo box.

Option Value	Meaning
Correct	Correct the input address. Do not generate suggestions. Generally used in batch mode.
Suggest	Generate suggestions (the default). Generally used by online applications where the operator can choose between a list of possibilities.
Complete	Use an incomplete address (fragment) to quickly generate suggestions. Used for online "fast completion" style of applications.
Certify	Use CASS certified validation rules defined by the USPS. Certify mode is only available for US addresses and requires additional database files to be installed in the DB directory.

### Archive Check

Archive Check option will include vanity names and outdated names (especially for localities) in the processing. Skipping this option will improve the speed very slightly, but may not correct addresses containing vanity names or outdated locality names. Two countries where you should definitely use this option are Germany and the US.

## Suggested Address Label Display



The address shown in the suggested address label display is formatted according to the address formatting rules in the country

## Address Result Panel Display

After parsing or validation, individual address fields are available for collection as part of a "suggestion". Suggestion 0 always holds the parsed fields. Suggestions numbered 1 and above hold the validated address fields. Individual fields are viewed in the List box in Output Results Frame. When the user has chosen the **UnFielded** and **validate** option, then Output frame list out the suggestion for the input address, once the user clicks on each suggestion the output address data from the List is mapped to corresponding fields in Fielded address display panel.

Organization	Building	Street_1	Street_2	HouseNumber	POBox	Locality	Province	ZipCode
MESTLINSTER		GOLDENWEST ST		14091		CA		92683
IDENTITY SYSTEMS		EAST PUTNAM ...		1445		OLD GREENWICH CT		06870
NORTH AMERICAN HE...		LEGACY DRIVE		5300		FLANO TX		75024
THE GRACE BUILDING		AVE OF AMERI...		1114		NEW YORK NY		10036
OFFICE OF REGULAT...		I (EYE) STE...		1634		WASHINGTON DC		20006-6083
RESEARCH TRIANGLE...		DEVELOPMENT ...		7001		F.O.BOX		13969
DIVISION GLOBAL S...		LEGACY DRIVE		6300		FLANO TX		75024
ERICSSON NETQUAL INC		ISAAC NEWTON...		1943		BOSTON VA		20190-5006
ERICSSON TREASURY...		LEGACY DRIVE		6300		FLANO TX		75024
ERICSSON WIRELESS...		WATERIDGE VI...		5012		SAN DIEGO CA		92121
CHEM USA CORPORATION		CENTRAL AVENUE		9445		NEWARK CALIFORNIA		94560
GRADUATE SCHOOL U...		UNIVERSITY A...		180		NEWARK		
DEPARTMENT OF PSY...						F.O. BOX		951563
BLOOM					P O BOX ...	CARMEL CA		93922
MONTANA STATE UNI...		WILSON HALL		2-260		BOZEMAN MT		59717-2400
ARIZONA STATE UNI...						TEMPE AZ		85287
DURG UNIVERSITY		OLD CHEM		223		F.O. BOX		90251
RUTGERS UNIVERSITY		WASHINGTON S...		111		NEWARK NJ		07102-3027
UNIVERSITY OF SOU...		WESTWOOD PLAZA		110		LCS ANGELES CA		90095-1461
CLARKSON UNIVERSITY					P O BOX ...	POTSDAM NEW YORK		13699-5705
LUKE'S MEDICAL CE...				RUSH-PRESBYTERIA...		1700 WEST YA...		470
OHIO STATE UNIVER...		NEIL AVE HALL		1827		COLUMBUS OH		43210
UNIVERSITY OF TIL...		EAST CAMPUS		603		CHAMBITEN		

## Validation Status and Database Version Display

Status bar displays Validation postal database version, AddressDoctor version and the status message when we press the validation button to validate the address.

Version: 5.0.2.335	DB Version: 5.00	DB Expiry: 28 02 2010
Status: Verified - Input data correct - all elements were checked and input matched perfectly		

Where version represents the Validation Database version and the status represent the Validation Status as mention in the following table for ASM using AddressDoctor v4:

Status Code	Meaning
0	Address is correct
1	Address was corrected
2	Needs correction; deliverability high
3	Needs correction; deliverability fair
4	Needs correction; deliverability small
5	Country not recognized
6	No valid country database found
7	Country not unlocked

Status Code	Meaning
8	No validate called yet
9	Insufficient information
10	No suggestions
11	Suggestions incomplete
12	Suggestions

Refer below table for validation status for ASM using AddressDoctor v5:

Status Code	Meaning
0	Verified - Input data correct - all elements were checked and input matched perfectly
1	Verified - Input data correct on input but some or all elements were standardised or input contains outdated names or exonyms
2	Verified - Input data correct but some elements could not be verified because of incomplete reference data
3	Verified - Input data correct but the user standardisation has deteriorated deliverability
4	Corrected - all elements have been checked
5	Corrected - but some elements could not be checked
6	Corrected - but delivery status unclear
7	Corrected - but delivery status unclear because user standardisation was wrong
8	Data could not be corrected completely, but is very likely to be deliverable - single match
9	Data could not be corrected completely, but is very likely to be deliverable - multiple matches
10	Data could not be corrected, but there is a slim chance that the address is deliverable
11	Data could not be corrected and is pretty unlikely to be delivered
12	FastCompletion Status - Suggestions are available - complete address
13	FastCompletion Status - Suggested address is complete but combined with elements from the input
14	FastCompletion Status - Suggested address is not complete
15	FastCompletion Status - Insufficient information provided to generate suggestions
16	Country recognized from ForceCountryISO3 Setting
17	Country recognized from DefaultCountryISO3 Setting

Status Code	Meaning
18	Country recognized from name without errors
19	Country recognized from name with errors
20	Country recognized from territory
21	Country recognized from province
22	Country recognized from major town
23	Country recognized from format
24	Country recognized from script
25	Country not recognized - multiple matches
26	Country not recognized
27	Parsed perfectly
28	Parsed with multiple results
29	Parsed with Errors - Elements change position
30	Parse Error - Input Format Mismatch
31	Validation Error: No validation performed because country was not recognized
32	Validation Error: No validation performed because required reference database is not available
33	Validation Error: No validation performed because country could not be unlocked
34	Validation Error: No validation performed because reference database is corrupt or in wrong format
35	Validation Error: No validation performed because reference database is too old - you need to contact AddressDoctor to obtain updated reference data

## Output Result Frame Column Selection Menu

Output Result Frame also provides option to select the list of column that user wants to view on the Output result panel. When the user right clicks on the list panel then popup menu displays on the screen showing the list of columns that user wants to enable or disable on the list.



**Score**

- ✓ Organization
- Department
- Nobility
- Title
- FirstName
- MiddleName
- LastName
- Function
- ✓ Building
- Sub-Building
- ✓ Street\_1
- ✓ Street\_2
- ✓ StreetNumber
- ✓ POBox
- DependentLocality
- ✓ Locality
- ✓ Province
- ✓ ZipCode
- Country
- Dbf Dep Locality
- MailSort
- County
- ZipPlus4
- Geocode Lat
- Geocode Long
- Geocode Unit
- Residue

Show details

Display CASS Fields

View CASS Report

Generate CASS Report

Clear CASS Report

## Field Status Display

Validation Status			
Field	Value	ResultStatus	MatchStatus
Organization	International School of Berne	empty	matched without errors
Department		empty	empty
Nobility		empty	empty
Title		empty	empty
FirstName		empty	empty
MiddleName		empty	empty
LastName		empty	empty
Function		empty	empty
Building		empty	empty
Sub-Building		empty	empty
Street_1	Mattenstutz	checked and corrected (changed or inserted)	not found
Street_2		empty	empty
HouseNumber		empty	empty
POBox		empty	empty
DependentLocality		empty	empty
Locality	MUENCHENBUCHSEE	checked and corrected (changed or inserted)	empty
Province	Bern	checked and corrected (changed or inserted)	empty
ZipCode	3053	checked and corrected (changed or inserted)	matched with errors
Country	SWITZERLAND	empty	empty

OK Cancel

### Match Status

All results share a Match Status that describes how the address elements matched to the postal reference data. Refer below table for match status for ASM using AddressDoctor v4:

Match Status	Meaning
0	Empty
1	Not found
2	Not Checked (no reference data or no chance of success)
3	Matched with errors
4	Matched without errors

Refer below table for match status for ASM using AddressDoctor v5:

val_status	Meaning
0	Empty
1	Not found
2	Not checked (no reference data)
3	Wrong - Set by validation only
4	Match with errors in this element
5	Match with changes
6	Match without errors

### Result Status

The Result Status indicates for each address component if and how it has been modified during the address validation process. Refer below table for result status for ASM using AddressDoctor v4:

Result Status	Meaning
0	Empty
1	Not checked
2	Not checked but standardized
3	Checked and corrected (changed or inserted)
4	Validated, but changed (synonyms, old names)

Result Status	Meaning
5	Validated, but standardized
6	Validated and unchanged

Refer below table for result status for ASM using AddressDoctor v5:

val_molds	Meaning
0	Empty
1	Not validated and not changed
2	Not validated bus standardized
3	Validated but not changed due to invalid input
4	Validated but not changed due to lack of reference data
5	Validated but not changed due to multiple matches
6	Validated and changed by eliminating the input value
7	Validated and changed due to correction based on reference data
8	Validated and changed by adding value based on reference data
9	Validated, not changed, but delivery status not clear
12	Validated, verified but changed due to outdated name
13	Validated, verified but changed from exonym to official name
14	Validated, verified but changed due to standardization based on casing or language
15	Validated, verified and not changed due to perfect match

## CASS Field Status Display

When validating address using validation mode as Certify, The Result panels shows validated address fields and when Display CASS Fields is selected will pop up CASS fields dialog as shown below:

CASS Fields	
Zip+4	1379
Delivery Point	99
Barcode	06870137999
Recordtype	H
Carrier Route	C006
Special Flag	B0
Congressional District Number	04
Delivery Point Check Digit	1
EWS Flag	N
Highrise Default	Y
Highrise Exact	N
Rural Route default	N
Rural Route Exact	N
LACS	
DPV Confirmation	
DPV CMRA	
DPV False Positive	
DPV FootNote1	AA
DPV FootNote2	
DPV FootNote3	

OK Cancel

## CASS Summary Report Display

USPS Form 3553 CASS certification can be generated by selecting **View CASS Report** from popup menu displayed in the address result tab for CASS certified address. Selecting this option displays **CASS Summary** report dialog as shown below:

Informatica Address Standardization Module Workbench - CASS Summary Report

**CASS Summary Report**

**A. Software**

1. CASS Certified Company Name AddressDoctor	2. CASS Certified Software Name & Version AddressDoctor v4.1.14.465	3. Configuration ABC
4. Z4 Certified Company Name N/A	5. Z4 Certified Software Name & Version N/A	6. Configuration N/A
7. DirectDPV Certified Company Name N/A	8. DirectDPV Certified Software Name & Version N/A	9. Configuration N/A
10. eLOT Certified Company Name N/A	11. eLOT Certified Software Name & Version N/A	12. Configuration N/A
1. MASS Certified Company Name N/A	2. MASS Certified Software Name & Version N/A	3. Configuration N/A
		4. NLOC Serial No. N/A

**B. List**

1. List Processor's Name AddressDoctor	2. Date List Processed Master File 09/02/2009	3. Date of Database Processed Used a. ZIP + 4 File 09/01/2009
	b. Z4Change N/A	b. Z4Change N/A
	c. DirectDPV N/A	c. DirectDPV N/A
	d. eLOT N/A	d. eLOT N/A
	e. CRIS N/A	e. CRIS N/A
4. List Name or ID No. D:\temp\TRF5888.mxa	Number of Lists 1	Total Records 1

**C. Output**

Output Rating	1. Total Coded	2. Validation Period From	To	Output Rating	1. Total Coded	2. Validation Period From	To
a. Zip + 4/DPV Confirmed	0	09/01/2009	09/30/2009	d. 5-Digit Coded	0	09/01/2009	09/30/2009
b. Z4Change Processed	0	N/A	N/A	e. CRK1 Coded	0	09/01/2009	09/30/2009
c. DirectDPV	0	N/A	N/A	f. eLOT Assigned	0	N/A	N/A

**D. Mailer**

1. I certify that the mailing submitted with this form has been coded (as indicated above) using CASS Certified software meeting all of the requirements listed in the DMM Section 700.

2. Name and Address of Mailer  
N/A

1. Mailer's Signature  
N/A

2. Date Signed  
09/02/2009 14:53:03

**E. Qualitative Statistical Summary (QSS)**

High Rise Default	High Rise Exact	RR Default	RR Exact	LACS	EDS	DATE
0	0	0	0	0	0	0

Generate CASS Report   Refresh CASS Summary   Clear CASS Summary   Close

## Statistics Reports - CASS Certification

ASM also provides options to generate USPS CASS 3553 Summary report which displays total records coded in each category. The report can be generated in HTML and XML format. CASS 3553 summary report sections are explained in detail under **CASS Summary Report Display**.

Generate CASS Report   Refresh CASS Summary   Clear CASS Summary   Close

## File Menu Options

### File | Clear Results

Menu option **File > Clear Results** menu option is used to clear the address result output display windows.

### File | Save Input

Menu option **File > Save Input** will prompt File Dialog, this options reads the file name from user and dumps address input and options(including validation mode, preload option, attributes) in the input file, which can be used as input for ASM Batch Test utility.

### File | View CASS Summary

Menu option **File > View CASS Summary** will display CASS Summary report dialog.

### File | Generate CASS Report

Menu option **File > Generate CASS Report** will prompt File Dialog, this options reads the file name from user and generate CASS summary for the input address. File format can be either XML or HTML.

### File | Clear CASS Summary

Menu option **File > Clear CASS Summary** is used to clear accumulated CASS summary after multiple validation using validation mode as certify.

### File | Exit

Menu option **File > Exit** will prompt to close and exit all SSA-NAME3 Workbench sessions.

## ASM Workbench and Batch Test utility

Using ASM Workbench, select **File > Save Input** option, the current Fielded or Unfielded Workbench input can be dumped to flat file, this flat file can be used as input for the ASM Batch Test utility `asmiss`.

This is a sample dump of UnFielded Address Input created by the ASM Workbench:

```
# ***Informatica's ASM input file***
# -hlocalhost:1666
# -dSwitzerland -cWIN1250 -l -mSuggest -v -yENGLISH -z -S -a -A -th
WOLFTRONIC Disco & Concert Equip.
zur Brunnenstube
Aeugst am Albis
CHE
```

This is a sample dump of Fielded Address Input created by the ASM Workbench:

```
# ***Informatica's ASM input file***
# -hlocalhost:1666
# -dSwitzerland -cWIN1250 -i -l -mSuggest -v -yENGLISH -z -S -a -A -th
00 International School of Berne
01
02
03
04
05
06
07
08
09
10 Mattenstutz
11
12
13
14
15 MUENCHENBUCHSEE
16 Bern
17 3053
18 Switzerland
#
```

The command to run the ASM Batch Test utility is as follows:

```
SSABIN%\asmiss [-h<host:port>] <ASM Workbench generated input file>
```

**Note:** The -h option must be specified either in the input file or on the command line. The command line overwrites any value specified in the input file.

# INDEX

%SSABIN% [353](#)

## A

Address Input [354](#)  
Address Standardization [330](#)  
AddressDoctor [330](#), [354](#)  
Archive Check [354](#)  
ASM Batch Test  
    SSABIN% [366](#)  
ASM Workbench [353](#), [366](#)

## B

Batch Mode [354](#)

## C

CASS [341](#)  
CASS Certification [354](#)  
CASS Summary [354](#)  
character set [330](#)  
Command line [353](#)  
Country Preload Option  
    Partial Preload  
    Full Preload [354](#)

Country Specific  
    AddressDoctor [354](#)

## I

Informatica Global Customer Support  
    contact information [16](#)  
initialize [330](#)  
Interactive Mode [354](#)

## P

Parsing [341](#), [354](#)

## S

Search Server Host [353](#)  
ssan3\_addr\_parse [341](#)  
ssan3\_addr\_set\_attrib [330](#)  
ssan3\_addr\_set\_lines [341](#)  
ssan3\_open [330](#)

## U

Unfielded Address [353](#)