

Unicenter[®] Service Desk

Modification Guide

r11.2



This documentation (the "Documentation") and related computer software program (the "Software") (hereinafter collectively referred to as the "Product") is for the end user's informational purposes only and is subject to change or withdrawal by CA at any time.

This Product may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA. This Product is confidential and proprietary information of CA and protected by the copyright laws of the United States and international treaties.

Notwithstanding the foregoing, licensed users may print a reasonable number of copies of the Documentation for their own internal use, and may make one copy of the Software as reasonably required for back-up and disaster recovery purposes, provided that all CA copyright notices and legends are affixed to each reproduced copy. Only authorized employees, consultants, or agents of the user who are bound by the provisions of the license for the Software are permitted to have access to such copies.

The right to print copies of the Documentation and to make a copy of the Software is limited to the period during which the license for the Product remains in full force and effect. Should the license terminate for any reason, it shall be the user's responsibility to certify in writing to CA that all copies and partial copies of the Product have been returned to CA or destroyed.

EXCEPT AS OTHERWISE STATED IN THE APPLICABLE LICENSE AGREEMENT, TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS PRODUCT "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO THE END USER OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS PRODUCT, INCLUDING WITHOUT LIMITATION, LOST PROFITS, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED OF SUCH LOSS OR DAMAGE.

The use of this Product and any product referenced in the Documentation is governed by the end user's applicable license agreement.

The manufacturer of this Product is CA.

This Product is provided with "Restricted Rights." Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7013(c)(1)(ii), as applicable, or their successors.

All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

Copyright © 2006 CA. All rights reserved.

CA Product References

This document references the following CA products:

- Advantage™
- Allfusion®
- BrightStor®
- CleverPath™ Portal
- eTrust® Embedded Identity and Access Management (eTrust eIAM)
- Unicenter® Asset Management
- Unicenter® Asset Portfolio Management (APM)
- Unicenter® Management Portal
- Unicenter® Network and Systems Management (NSM)
- Unicenter® Remote Control
- Unicenter® Service Desk
- Unicenter® Knowledge Tools (KT)
- Unicenter® Software Delivery
- Unicenter® TNG for Windows

Contact Technical Support

For online technical assistance and a complete list of locations, primary service hours, and telephone numbers, contact Technical Support at
<http://ca.com/support>.

Contents

Chapter 1: Introduction	21
Audience	21
What You Need to Know	21
Adapting Unicenter Service Desk	22
Chapter 2: Customizing Notification Methods	25
The Notification Process	26
Notification Method Variables	26
Basic Environment Variables	27
Attribute Variables	30
The Notification File	31
How Environment Problems are Overcome	32
Customization Steps	33
Create a Script	33
Add the Notification Method (Web Client Administration Interface)	34
Add the Notification Method (UNIX)	35
Example Syntax	36
Chapter 3: Custom Reports	37
How to Design Reports	38
Selecting Information for the Report	38
How to Create a Report Template	39
How To Generate Reports	47
The Report Command	48
How to Display a Dialog (UNIX Only)	48
Report Template Reference	49
Report Template Variable Expressions	49
Report Template Functions	52
Report Template BLOCK Statements	55
Report Template FOOTER Statements	57
Report Template HEADER Statements	58
Report Template HEADER2 Statements	59
Report Template PAGE FOOTER Statements	60
Report Template PAGE HEADER Statements	61
Report Template PRINT Statements	62
Customize Crystal Reports	64

Chapter 4: Customization of Queries and Messages	65
Scoreboard Queries	65
Stored Queries for Logged in User	66
Time-Based Queries.....	70
ITIL-Specific Queries.....	73
Activity Notification Messages Customization	73
Formatting Attributes for Activity Notifications	74
Attributes from the Activity Log Object	77
Information on Specific Requests	78
Information on Specific Change Orders	79
Chapter 5: Customizing the Schema Using the Web Screen Painter	81
Schema Designer Modification Overview	82
The WSP Schema Designer Tool	83
Schema Designer Tabs	84
Table Info Tab	84
Column Info Tab.....	86
Advanced Tabs	89
Schema Designer Tasks	90
Modify Tables or Columns	90
Add a New Table.....	90
Add a New Column.....	91
Save Changes	91
Test Schema Modifications	92
Revert Schema Modifications	93
Publish Schema Modifications	94
Test to Production Migration	95
Change or Delete Site-Defined Columns after Publishing.....	96
Chapter 6: Customizing the Web Interface	99
Web Screen Painter (WSP)	101
Start Web Screen Painter.....	102
Open a Form for Editing	103
Create a New Form	104
Form Edit Window	105
Edit List and Detail Forms in Design View	106
Edit in Source View	118
Edit Menu Bars	119
Edit Stylesheets	121
Edit HTML and JavaScript.....	123

Saving Changes	123
Deleting Changes Prior to Publication.....	124
Deleting Forms after Publication	124
Publishing Changes	124
Test to Production Migration	125
HTML Templates (HTMPL Form).....	126
Overview	126
Template Naming Conventions.....	126
HTMPL Directories	127
Creating Web Form Groups	130
HTMPL Tags	131
PDM_EVAL: Insert the Value of a Pre-Processor Variable.....	131
PDM_FORM: Start an HTML Form with a Session ID	132
PDM_FMT: Format Text from a Server Variable	133
PDM_IF: Conditional Processing.....	136
PDM_INCLUDE: Inserting from a Different File	138
PDM_JSCRIPT: Conditionally Include a JavaScript File	140
PDM_LINK: Create a Hyperlink Invoking an HTMPL Operation	141
PDM_LIST: Format a List of Database Rows	142
PDM_MACRO: Insert Text from a Macro File	146
PDM_NOTEBOOK: Create a Notebook	151
PDM_PRAGMA: Specify Server Information	152
PDM_SCOREBOARD: Build a Scoreboard Tree.....	153
PDM_SET: Set the Value of a Server Variable	154
PDM_TAB: Create a Tab within a Notebook	154
PDM_WSP: Control WSP Preview	155
Server Variables	156
Simple Variables	156
Property Variables	157
Environment Variables	159
Business Object Variables	160
List Variables	162
Server Operations	163
Supported Operations	163
Operation Variables	166
Syntax of PRESET, PRESET_REL, ALG_PRESET, and ALG_PRESET_REL	171
Link Examples	172
Advanced Customization	172
The Web Engine and Its Cache.....	173
The pdm_webcache Utility	174
How to Modify HTML Templates	175
Files That Should Not be Modified	176

Guidelines for New HTMPL Files	177
How to Add User Defined State Information	178
How to directly create a Request from a Template	179
Directories Used by Your HTTP Server	179
Looking Up Information in Reference Tables	181
Web Engine PreProcessing	183
Free-Form Customization of Detail Forms	185
Understanding List Forms	191
Customizing the Edit in List Feature	193
Integrating with Your Own Web Pages	195
Customizing JavaScript	198
Updating and Creating Change Orders as Employee User	202
Add "Closed Change Orders" link to the Employee Scoreboard	203
Download Attachments	204
 Chapter 7: The Screen Painter	 205
The Issue Management Function	205
Configuration Overview	206
Database Schema Customization	207
Object Schema Customization	207
Forms Customization	208
Form Naming Conventions	210
How to Customize Forms	211
Start Screen Painter	212
Controls	212
Add a Control	213
Functionality Restrictions	213
Screen Painter User Access	213
pdm_manage_user_forms Utility	214
 Appendix A: Data Element Dictionary	 215
Access_Levels Table	215
Access_Type Table	216
Act_Log Table	220
Act_Type Table	221
Act_Type_Assoc Table	225
Active_Boolean_Table Table	226
Active_Reverse_Boolean_Table Table	227
admin_tree Table	227
Am_Asset_Map Table	228
Animator Table	229

Archive_Purge_History Table	230
Archive_Purge_Rule Table	231
Asset_Assignment Table	232
Atomic_Condition	232
Attached_Events Table	234
Attached_SLA Table	235
Attachment Table	236
Attachment_Lrel Table	238
atmnt_folder Table	238
Attribute_Name Table	239
Audit_Log Table	240
Behavior_Template Table	241
Boolean_Table Table	242
Bop_Workshift Table	243
BU_TRANS Table	244
Business_Management Table	244
Business_Management_Class Table	245
Business_Management_Repository Table	246
Business_Management_Repository_Lrel Table	247
Business_Management_Status Table	248
ca_asset_type Table	248
ca_company Table	249
ca_company_type Table	251
ca_contact Table	252
ca_contact_type Table	255
ca_country Table	256
ca_job_function Table	257
ca_job_title Table	258
ca_location Table	259
ca_model_def Table	262
ca_organization Table	264
ca_owned_resource Table	266
ca_resource_class Table	271
ca_resource_cost_center Table	272
ca_resource_department Table	273
ca_resource_family Table	274
ca_resource_gl_code Table	275
ca_resource_operating_system Table	276
ca_resource_status Table	277
ca_site Table	278
ca_state_province Table	279
Call_Req Table	280

Call_Req_Type Table	284
Call_Solution Table	285
Change_Act_Log Table	286
Change_Category Table	288
Change_Request Table	289
Change_Status Table	294
Chg_Template Table	295
Chgcat_Group Table	296
Chgcat_Loc Table	297
Chgcat_Workshift Table	297
CI_ACTIONS Table	298
CI_ACTIONS_ALTERNATE Table	298
CI_BOOKMARKS Table	299
CI_DOC_LINKS Table	300
CI_DOC_TEMPLATES Table	300
CI_DOC_TYPES Table	301
CI_PRIORITIES Table	301
CI_STATUSES Table	301
CI_WF_TEMPLATES Tables	302
Column_Name Table	302
Contact_Method Table	303
Controlled_Table Table	304
Cr_Call_Timers Table	305
Cr_Status Table	306
Cr_Stored_Questions Table	307
Cr_Template Table	308
D_PAINTER Table	309
Delegation_Server Table	310
Document_Repository Table	311
Domain Table	312
Domain_Constraint Table	313
Domain_Constraint_Type Table	314
EBR_ACRONYMS Table	315
EBR_DICTIONARY Table	315
EBR_DICTIONARY_ADMIN Table	316
EBR_FULLTEXT Table	316
EBR_FULLTEXT_ADMIN Table	317
EBR_FULLTEXT_SD Table	318
EBR_FULLTEXT_SD_ADMIN Table	319
EBR_INDEX Table	320
EBR_INDEX_ADMIN Table	321
EBR_INDEXING_QUEUE Table	321

EBR_KEYWORDS Table	322
EBR_LOG Table	322
EBR_METRICS Table	324
EBR_NOISE_WORDS Table	324
EBR_PATTERNS Table	325
EBR_PREFIXES Table	325
EBR_PROPERTIES Table	326
EBR_SUBSTITS Table	326
EBR_SUFFIXES Table	327
EBR_SYNONYMS Table	327
EBR_SYNONYMS_ADM Table	327
ES_CONSTANTS Table	328
ES_NODES Table	328
ES_RESPONSES Table	330
ES_SESSIONS Table	330
Event_Delay Table	331
Event_Delay_Type Table	332
event_log Table	333
event_type Table	333
Events Table	334
ext_appl Table	335
External_Entity_Map Table	336
Form_Group Table	337
Global_Change_Extension Table	338
Global_Change_Queue Table	339
Global_Contact Table	340
Global_Issue_Extension Table	341
Global_Issue_Queue Table	342
Global_Location Table	343
Global_Organization Table	344
Global_Product Table	345
Global_Queue_Names Table	345
Global_Request_Extension Table	346
Global_Request_Queue Table	347
Global_Servers Table	348
Global_Table_Map Table	350
Global_Table_Rule Table	350
Group_Loc Table	351
Group_Member Table	352
Impact Table	353
INDEX_DOC_LINKS Table	353
Interface Table	354

ISS_Template Table	355
Isscat_Group Table	356
Isscat_Loc Table.....	356
Isscat_Workshift Table	357
Issue Table	357
Issue_Act_Log Table	362
Issue_Category Table.....	364
Issue_Property Table	365
Issue_Status Table	366
Issue_Workflow_Task Table	367
KD_ATTMNT Table	370
kdlinks Table	370
Key_Control Table	371
Knowledge_Keywords Table	371
Knowledge_Lrel_Table Table	372
KT_REPORT_CARD Table	372
LONG_TEXTS Table	374
Lrel_Table Table	374
Managed_Survey Table	375
Mgs_Act_Log Table	377
Mgs_Status Table.....	378
Note_Board Table	379
NOTIFICATION Table	380
Notification_Urgency Table	381
Notify_Log_Header Table.....	381
Notify_Object_Attr Table	383
NR_Comment Table	383
O_COMMENTS Table	384
O_EVENTS Table	385
O_INDEXES Table	386
Options Table	387
P_GROUPS Table	388
Pcat_Group Table.....	388
Pcat_Loc Table	389
Pcat_Workshift Table	389
Person_Contacting Table	390
Priority Table	391
Prob_Category Table.....	392
Product Table	393
Property Table.....	394
Property_Template Table	395
Queued_Notify Table.....	396

Quick_Template_Types Table	397
Remote_Ref Table	398
Reporting_Method Table	399
Req_Property Table	399
Req_Property_Template Table	400
Response Table	401
Reverse_Boolean_Table Table	402
Rootcause Table	403
Rpt_Meth Table	404
SA_Policy Table	405
SA_Prob_Type Table	406
Sequence_Control Table	407
Server_Aliases Table	408
Server_Zones Table	409
Service_Contract Table	410
Service_Desc Table	411
session_log Table	412
session_type Table	413
Severity Table	414
SHOW_OBJ Table	414
SKELETONS Table	415
SLA_Contract_Map Table	419
SLA_Template Table	419
Spell_Macro Table	420
Spell_Macro_Type Table	422
SQL_Script Table	423
Survey	423
Survey_Answer	424
Survey_Answer_Template	425
Survey_Question Table	426
Survey_Question_Template Table	427
Survey_Stats Table	428
Survey_Template Table	429
Survey_Tracking Table	431
Table_Name Table	432
Task_Status Table	432
Timespan Table	434
Timezone Table	435
Transition_Points Table	437
Task_Type Table	437
Type_Of_Contact Table	438
Urgency Table	439

User_Query	440
usp_contact Table	441
usp_organization Table	443
usp_owned_resource Table.....	444
USP_PREFERENCES Table	446
USP_PROPERTIES Table.....	450
Wftpl_Group Table	451
Workflow_Task Table	451
Workflow_Task_Template Table	454
wspcol Table	456
wsptbl Table	458

Appendix B: Objects and Attributes 461

acc_lvls Object	461
acctyp Object	462
act_type_assoc Object	464
actbool Object.....	465
actrbool Object	465
ADMIN_TREE Object	466
alg Object.....	467
am_asset_map Object	468
ANI Object	469
arcpur_hist Object	470
arcpur_rule Object.....	471
atev Object	472
atomic_cond Object.....	473
attached_sla Object	474
attmnt Object	475
attmnt_folder Object.....	476
attmnt_lrel Object	477
aty Object	478
audlog Object	480
bhvtpl Object.....	481
bmcls Object	482
bmhier Object	483
bmlrel Object.....	484
bmrep Object	484
bms Object	485
bool Object.....	486
BU_TRANS Object	486
ca_cmpny Object	487
chg Object	489

chg_tpl Object	492
chgalg Object	493
chgcat_grp Object	494
chgcat Object	494
chgcat_loc Object	496
chgcat_workshift Object	496
chgstat Object	497
CI_ACTIONS Object	498
CI_ACTIONS_ALTERNATE Object	499
CI_BOOKMARKS Object	499
CI_DOC_LINKS Object	500
CI_DOC_TEMPLATES Object	501
CI_DOC_TYPES Object	501
CI_PRIORITIES Object	502
CI_STATUSES Object	502
CI_WF_TEMPLATES Object	503
cmth Object	504
cnote Object	505
cnt Object	506
cost_cntr Object	509
country Object	510
cr Object	511
cr_prp Object	513
cr_prptpl Object	514
cr_tpl Object	515
crs Object	516
crsol Object	517
crsq Object	518
crt Object	519
ctab Object	520
ctimer Object	520
ctp Object	521
dcon Object	522
dcon_typ Object	523
dept Object	523
dlgsvr Object	524
dmn Object	525
doc_rep Object	526
EBR_ACRONYMS Object	527
EBR_FULLTEXT Object	527
EBR_FULLTEXT_ADMIN Object	529
EBR_FULLTEXT_SD Object	530

EBR_FULLTEXT_SD_ADMIN Object	531
EBR_INDEXING_QUEUE Object	532
EBR_KEYWORDS Object	533
EBR_LOG Object	533
EBR_METRICS Object	535
EBR_NOISE_WORDS Object	535
EBR_PATTERNS Object	536
EBR_PREFIXES Object	536
EBR_PROPERTIES Object	537
EBR_SUBSTITS Object	538
EBR_SUFFIXES Object	538
EBR_SYNONYMS Object	539
EBR_SYNONYMS_ADMIN Object	539
ES_CONSTANTS Object	540
ES_NODES Object	540
ES_RESPONSES Object	542
ES_SESSIONS Object	542
event_log Object	543
event_type Object	544
evt Object Object	545
evtdly Object	546
evtdlytp Object	547
ext_entity_map Object	547
fmgrp Object	548
g_chg_ext Object	549
g_chg_queue Object	550
g_cnt Object	551
g_cr_ext Object	552
g_cr_queue Object	553
g_iss_ext Object	554
g_iss_queue Object	555
g_loc Object	556
g_org Object	557
g_prod Object	558
g_qname Object	558
g_tblmap Object	559
g_tblrule Object	560
g_srvrs Object	561
gl_code Object	562
grc Object	563
grp_loc Object	564
grpmem Object	564

hier Object	565
imp Object	566
INDEX_DOC_LINKS Object	567
intfc Object	567
iss Object	568
iss_prp Object	571
iss_tpl Object	572
iss_wf Object	573
issalg Object	575
isscat Object	576
isscat_grp Object	577
isscat_loc Object	578
isscat_workshift Object	578
issstat Object	579
job_func Object	580
KCAT Object	581
KD Object	582
KD_ATTMNT Object	585
kdlinks Object	586
kmlrel Object	586
KT_REPORT_CARD Object	587
kwrd Object	588
loc Object	589
LONG_TEXTS Object	591
lr Object	591
lrel1 Object	593
macro Object	593
macro_type Object	594
mfrmod Object	595
mgs Object	597
mgsalg Object	598
mgsstat Object	599
NOTIFICATION Object	600
notque Object	600
noturg Object	601
nr Object	602
nr_com Object	606
nrf Object	607
ntfl Object	608
O_COMMENTS Object	609
O_EVENTS Object	610
opsys Object	611

options Object	612
org Object	613
P_GROUPS Object	615
pcat Object	615
pcat_grp Object	616
pcat_loc Object	617
pcat_workshift Object	618
perscnt Object	618
position Object	619
pri Object	620
prod Object	621
prp Object	621
prptpl Object	622
quick_tpl_types Object	623
rc Object	624
response Object	625
rev_bool Object	626
rptm Object	626
rptmeth Object	627
rrf Object	628
rss Object	629
sapolicy Object	630
saprobtyp Object	631
sdsc Object	632
sdsc_map Object	633
seq Object	634
session_log Object	635
session_type Object	635
sev Object	636
SHOW_OBJ Object	637
site Object	637
slatpl Object	638
srvr_aliases Object	639
srvr_zones Object	640
state Object	640
survey Object	641
svc_contract Object	642
svy_atpl Object	643
svy_ans Object	644
svy_qtpl Object	645
svy_ques Object	646
svy_tpl Object	647

svystat Object	648
svytrk Object	648
tskstat Object	649
tsky Object	650
tspan Object	651
typecnt Object	652
tz Object	653
urg Object	654
USP_PREFERENCES Object	655
USP_PROPERTIES Object	659
usq Object	659
vpt Object	660
wf Object	661
wftpl Object	662
wftpl_grp Object	664
wrkshft Object	664
wspcol Object	665
wsptbl Object	666

Appendix C: Table and Object Cross-References 669

Table to SQL Name and Object	669
SQL Name to Table and Object	677
Object to Table and SQL Name	685

Appendix D: Schema Files Syntax 695

TABLE Statement	696
TABLE_INFO Statement	700
Mapping Statement	702

Appendix E: Object Definition Syntax 705

Directories	705
Types of Statements	705
OBJECT Statement	706
ATTRIBUTES Optional Statement	708
FACTORY Optional Statement	712
STANDARD_LISTS Optional Statement	714
MODIFY Statement	717
MODIFY FACTORY Statement	718

Chapter 1: Introduction

Welcome to Unicenter Service Desk, the most advanced tool for delivering superior user support services in heterogeneous UNIX and Windows computing environments. Unicenter Service Desk is a comprehensive, integrated solution for the total automation and management of both external (customer support) and internal (enterprise service center management) service desks.

Audience

This guide explains how to customize Unicenter Service Desk. It is intended for the administrator or application developer whose job it is to change the way Unicenter Service Desk looks and works. Some customization techniques apply to all of Unicenter Service Desk, while other techniques are specific to each function.

What You Need to Know

To fully understand the information in this guide, you should have a basic knowledge of:

- Programming terminology, techniques, and concepts
- The operating system with which you will be working
- Basic database concepts such as records (rows), fields (columns), tables (relations), schemas, and SQL
- Object-oriented programming concepts, if you are customizing windows and the database schema

Before reading this guide, you should be familiar with the information in the Unicenter Service Desk *Implementation Guide* and *Administrator Guide*. This guide does not describe any of the basic concepts that those guides discuss.

Adapting Unicenter Service Desk

The Unicenter Service Desk product line is an exceptionally flexible product designed to fulfill a variety of IT Service Management functions. We strive to provide an out-of-the-box product experience that includes a broad feature set as well as a variety of best practice content to ensure that your service management needs are met as rapidly, and fully, as possible.

While we believe that the default implementation of the Unicenter Service Desk product will match up closely with the processes and terminology used in most IT organizations, we recognize the need to extend the product to embrace the unique aspects of your operation. To that end, the Unicenter Service Desk system includes a broad spectrum of approaches to adapt the system to meet your unique needs: end-user personalization, system-wide configuration, tool-based adaptation, even code-level customization.

This manual describes the different types of approaches that are available and helps you understand which approach, or approaches, will work best to make the most out of your system. It also provides detailed instructions and guidelines for using the provided tools and methods.

A wide variety of terms have been used to describe extending the functionality of software products. Probably the most overused term is "customize", which is frequently used to encompass any activity that changes the behavior of the product. Recent changes in the industry indicate an improved awareness that different types of "customizations" have different benefits and/or consequences. So for the sake of clarity, we have provided the following definitions that can be applied to approaches used to extend the Unicenter Service Desk product line:

Personalization

This allows the behavior of the product to be modified for each individual end-user. Typically, personalization takes the form of how screen elements are arranged, if a feature is available, or what data is shown in certain views. The changes are usually implemented by the end-user by setting options (usually via easy-to-use dialogs). These personalization definitions normally are stored with the user's profile information in the database and will be automatically migrated when a system is upgraded to a later version. Unicenter Service Desk also provides additional "personalization" capabilities that may be associated with end-user's associated with a "group" to simplify administration and security.

Configuration

This provides system-wide changes in behavior and may be used to set the basic operating parameters of the product. Configuration is normally performed initially after the product has been installed to identify DBMS access information, initial user definitions and passwords, and other integration points necessary to run the system. Some systems, like Unicenter Service Desk, allow re-configuring the system at later times as well. Configuration information may or may not be stored in the database and migrated when upgrading to later version of the product.

Adaptation

This provides system-wide changes in behavior and is accomplished using tools and methods provided with the product. The changes to the base product are stored and/or organized in a manner that allows the changes to be "migrated" to a later version of the base product without loss of the site-defined behavior. The majority of the approaches described in this manual fall into this category and this is the recommended approach for fulfilling needs that are unique to your operation or business. Since adaptations are performed as part of the normal operation of the product, these changes are part of the supported solution.

Being able to easily retain your personalization and configuration parameters plus your, adaptations when upgrading to later releases of the base product is an important, and somewhat unique, feature of the Unicenter Service Desk product line. This is provided to allow you to adapt the system to your unique processes and terminology without concerns that it will be difficult or costly to retain the changes that you need.

Customization

This provides system-wide changes in product behavior by altering exposed sections of the product "source code" using programming tools. Because these changes are outside of the "awareness" of the application and are, in fact, changes to the originally delivered files, great care must be taken to preserve a customization during routine maintenance activities. When new versions of the product (even patches) are installed, new files are delivered that may overwrite the "customized" versions or the altered sections of source code may no longer be valid within the new product version. "Customized" aspects of the system are not normally supported as part of the normal product support contract and may prevent the support organization from being able to troubleshoot problems in other areas of the product as well. Customizing the source code of the system is not recommended. In the rare cases where customization is necessary, only qualified CA CATS personnel should perform the customization. The CA CATS organization is capable of providing the change, as well as establishing a support agreement that ensures that your customized behavior can be supported.

Chapter 2: Customizing Notification Methods

The Unicenter Service Desk automatic notification methods notify personnel at key points in the service desk management process. The standard notification methods are shown as follows (see the online help for a complete description of each):

- Email
- FAXserve
- Notification (Log)
- Pager_Email

You can define customized notification methods to specify a new method of transmission, for example, voice mail, display boards, or a specific printer. You can also access data from another application and include it in the notification message.

This chapter explains how to create your own customized notification methods. For more information about using the standard notification methods, see Notifications in the chapter “Policy Implementation” of the *Administrator Guide*.

The Notification Process

Ticket notifications (applicable to issues, change orders, and requests) are processed when the ticket is saved, as described by the following:

- If the notification method is other than Notification (for example, Email or FAXserve), the notification processor executes the notification method for each contact in the list. This method is typically an executable or shell script, which is launched in a new process. Details about the notification are stored in environment variables for easy access by the executable/script.
- For each notification requested, the notification processor sets the NX_NTF_MESSAGE and NX_NTF_SUMMARY environment variables using the Notification Message Title and Notification Message Body information provided on the Message Template notebook page of the Activity Notifications Detail window. If the recipient is a valid contact, additional environment variables (see page 27) are created using information in their Contact Detail record.
- If the Write To File (see page 30) option is selected for the notification, a text file is created with additional information that the notification method can use to obtain more detailed information.
- A list of contacts to receive the notification is built from the information provided on the Objects, Contacts, Types, and Survey notebook pages of the Activity Notifications Detail window. For those having a notification method matching the Notify Level and the log_all_notify Options Manager option installed, a notification is generated first to the notification log.

Notification Method Variables

Two sets of variables are created and made available to the notification method.

Basic Environment Variables

The first set of variables is created for every notification sent, independent of whether you select the Write To File option for the notification. They are written to the environment as environment variables that can be accessed by the notification method in the standard way. If you choose the Write To File option for the notification method, these variables are also written to the notification file (see page 31) in the notification section.

The following environment variables give you basic information about the notification. They are always defined, even if the corresponding value is empty:

Environment Variable	Description
NX_NTF_MESSAGE	The completed message template text, including full expansion of all variables
NX_NTF_SUMMARY	The completed message template header, including full expansion of all variables
NX_NTF_URGENCY	The notification urgency (1 is low, 4 is emergency)

The following environment variables are created only if the recipient is a valid Unicenter Service Desk contact, in which case they are set using values from the recipient's Contact Detail record as shown in the following table:

Variable	Contact Detail Window Fields
NX_NTF_BEEPER_PHONE	Pager Number
NX_NTF_COMBO_NAME	Last Name, First Name, Middle Name
NX_NTF_CONTACT	Contact ID
NX_NTF_EMAIL_ADDRESS	Email or Pager Email Address (depending on notification type)
NX_NTF_FAX_PHONE	Fax Number
NX_NTF_PUBLIC_PHONE	Phone Number
NX_NTF_USERID	System Login
NX_NTF_VOICE_PHONE	Alt. Phone #

Note: These variables are not created if the corresponding values are empty (with the noted exception of NX_NTF_CONTACT, which cannot be empty).

The environment variable @NX_EVENT_LOG_EXCLUDE defines a list of events that will not be written to the event log. List items are separated by commas (for example, @NX_EVENT_LOG_EXCLUDE = FAQ,KD_OPEN). Using the LOGIN,LOGOUT events from the following table, for example, @NX_EVENT_LOG_EXCLUDE value of LOGIN,LOGOUT will affect the application to not record login and logout events.

Event	Enum	By	Sets	Comments
LOGIN	1	SD		Specifies that the User logs in.
LOGOUT	2	SD	numdata1	Specifies that the User logs out, where numdata1=logout reason: 0=normal 1=timeout 2=abnormal)
CR_CREATE	3	SD	sd_obj_type, sd_obj_id, kd, numdata1	Indicates that the User creates a request, where numdata1=id of affected end user.
ISS_CREATE	4	SD	sd_obj_type, sd_obj_id, kd, numdata1	Indicates that the User creates a change order, where numdata1=id of the affected end user.
CHG_CREATE	5	SD	sd_obj_type, sd_obj_id, kd, numdata1	Indicates that the User creates an issue, where numdata1=id of the affected end user.
EMAIL	6	KT	kd	Specifies that the Analyst emails a document.
LINK	7	KT	kd, sd_obj_type, sd_obj_id	Indicates that the User accepts a solution, and links it to a ticket.
UNLINK	8	SD	sd_id, sd_obj_type, sd_obj_id	Specifies that the User unlinks a solution from a ticket.

Event	Enum	By	Sets	Comments
SEARCH	9	KT	numdata1,	Indicates that the User searches knowledge, where numdata1= CI_ASKED_QUES id.
FAQ	10	KT	numdata1	Indicates a FAQ search, where numdata1= O_INDEXES id (category).
DT_NAVIGATE	11	KT	kd, numdata1, textdata1	Indicates that the User navigates a decision tree, where numdata1= ES_NODES ID textdata1=path.
KD_BOOKMARK	12	KT	kd	Indicates that the User bookmarks a KD.
KD_COMMENT	13	KT	kd, numdata1	Indicates that the User adds a comment to a KD, where numdata1= O_COMMENTS id.
KD_CREATE	14	KT	sd_ob_type, sd_obj_id, kd	Specifies that a User creates a new document. SD IDs are used when a KD is created using submit knowledge from a request or an issue.
KD_OPEN	15	KT	kd, numdata1	Indicates that a User opens a KD, where numdata1=BU_TRANS ID.
KD_RATE	16	KT	Kd, numdata1	Indicates that a User rates a KD, where numdata1=BU_TRANS ID.
KD_NEW	17	KT	numdata1	Specifies that a User clicks on the New Documents folder in the Knowledge tab.

Attribute Variables

The second set of variables, called attribute variables, is available only if you choose the Write To File option when you define the notification method as described in The Notification File (see page 31) in this chapter. They are written to the notification file only—not to the environment. They are of the form:

`NX_NTF_attribute[.secondary_attribute]=value`

where:

attribute

The name of the attribute whose value you want to obtain. This is the attribute name as defined for the object. The appendix “Objects and Attributes” (see page 461) lists the complete list of all attribute names for any object. The most common objects associated with notifications are the ticket, which has an object name dependent of the type of ticket (for example, cr for requests), and the contact identifying the recipient, which has an object name of cnt. For example, the environment variable for the description attribute of a ticket might look as follows in the notification file:

`NX_NTF_DESCRIPTION=This is a sample description.`

secondary_attribute

If the first *attribute* is an internal identifier for another object, a secondary attribute is often attached to give more meaningful information using the dot notation. In database terms, attribute is a foreign key that points to a row in another table, rather than a simple data value. Using this raw key value would probably have little meaning. To save you the effort, many of these types of fields are resolved or de-referenced for you. When this is the case, *secondary_attribute* will be the value in the referenced table. For example, instead of writing the value for the assignee attribute, which is actually stored as the unique ID of the contact record for the assignee, the assignee's combined name is written by referring to the *combo_name* attribute for the contact object, as shown in this example:

`NX_NTF_ASSIGNEE.COMBO_NAME=Armstrong, Beth`

If an attribute does not have a value, the corresponding value is usually (NULL) or blank. For example:

`NX_NTF_CALL_BACK_DATE=(NULL)`

`NX_NTF_GROUP.COMBO_NAME=`

Note: An attribute variable that exists for both the ticket and the recipient is `NX_NTF_ID` (the `id` attribute), which is the unique database ID for the object.

The Notification File

If you check the Write To File option when you define a notification method, all the variables as listed in Basic Environment Variables (see page 27) and Attributes Variables (see page 30) in this chapter are written to a text file, which is closed prior to executing the notification method script or program. This notification file is written every time the notification method is invoked for a contact, and is a handy mechanism for passing relevant information to the notification script that is not otherwise available in the environment.

The full path of the notification file is set in the NX_NTF_FILENAME environment variable, which is available to the notification method's process. The file name is also added to the end of value you enter in the Notification Method field when defining the notification method. For example, if the Notification Method is 'pdm_perl -w mymethod.pl', then the actual process will execute 'pdm_perl -w mymethod.pl *unique_notification_file_name*'.

Important! It is up to the administrator to clean up the notification files. This is especially important for a site using a high volume of notifications, which can result in thousands of notification files a day. The files are located in the standard temporary directory (TEMP on Windows and TMP on UNIX). One suggestion is to delete the file at the end of the notification method script/program.

The notification file is a standard text file that is divided into sections. Each line contains either an attribute/value pair or a section marker. There are three sections in each notification file, as described by the following. All sections begin with "-----" followed by a new line.

SECTION=obj, where obj identifies the object type of the ticket

Iss

This gives information about the issue.

Chg

This gives information about the change order.

Cr

This gives information about the request.

SECTION=cnt

This gives information about the recipient.

SECTION=notification

This gives the same information that is available from the basic environment variables (see page 27).

Note: The section names for the ticket and recipient are actually the object names for the attributes in that section. The appendix "Objects and Attributes" (see page 461) lists the complete list of all attribute names for any object.

Several lines of attribute/value pairs, each of which represents an attribute of the corresponding object, are contained in each section. The Attribute Variables in this chapter provides the detailed information about how these lines are formatted and what they mean.

If an attribute value has line breaks within it, these are reproduced as new lines in the notification file. Therefore, the only key new lines that should interest your notification method process are the attribute/value lines (which always begin with NX_NTF) and the section markers, as described earlier. Before attempting to work with a notification file in your notification method process, you should generate a sample file and look at its contents.

How Environment Problems are Overcome

Most notification methods invoke an executable or shell script to read the environment variables and send the message. This works well on most UNIX servers, but difficulties arise reading the environment variables on a Windows server.

You can use a Perl script to overcome environment problems on Windows. Unicenter Service Desk includes a ready-to-use installation of the Perl interpreter called pdm_perl. Any Perl script invoked with pdm_perl as a notification method can reliably obtain the environment variables. The Perl script can read and format the environment variable values and carry on with the rest of the notification, such as invoking a pager or sending an email.

For Windows based servers, consider using the launchit utility. One of the functions of this utility is to invoke your scripts or programs in a shell environment similar to the Command Prompt with the proper environment variables set.

For example, if you write a Perl script named read_env.pl to read several of the environment variables described here, you can invoke it for a notification by entering the following in the Notification Method field on the Notification Method Detail window:

```
pdm_perl script_path/read_env.pl
```

This notification method will start up the Perl interpreter and execute the instructions in read_env.pl script.

Customization Steps

Creating a customized notification method involves the following steps:

1. Create a script (see page 33) to process the message template and transmit it to the recipient. The script can be any executable, depending on the platform. Third-party or public domain interpreters can also be used. Typically, Bourne shell scripts are used on UNIX and .bat files are used on Windows. If your script requires a special template, you must create it.
2. Add the new notification method (see page 34) to your site using the Unicenter Service Desk Web Client Administration Interface.

Create a Script

Use the following steps to create a notification method script:

1. Determine how you want the notification to be delivered (for example, printed on a particular printer).
2. Determine the contents of the notification message.
3. Specify what information from the message template to include in the notification.
4. Set up a script to transmit the notification.
5. Place the script in an executable file in the path of the Unicenter Service Desk server.

Add the Notification Method (Web Client Administration Interface)

After you create a script, you must define the new notification method to Unicenter Service Desk. To do so, perform these steps using the Administration Interface of the Web Client:

1. Choose Notification Methods from Notifications on the Administration Interface.

The Notification Method List appears.

2. Click the Create New button.

A Create New Notification Method window appears.

3. Enter data in the following fields:

Symbol

Identify the notification method. This is a required field.

Write to File

Check this box to write the context information of the notification method to a file.

Description

Describe the notification method.

Notification Method

Specify the full path of the executable script for the notification method. If the script or program can be resolved using the system path, the full path need not be specified. For a Windows Server, consider using the launchit.exe utility to invoke your script or program. See the online help for more information on the launchit utility.

Note: Since the notification method runs from the Unicenter Service Desk server, you must put the notification method script in a directory that can be accessed from the path on the server or specify the full path to the script. On UNIX, depending on the shell you are running, you may be able to check this by executing this command:

which pathname_to_script

If there appears to be a problem with the notification methods, examine the logs in the \$NX_ROOT/log directory on UNIX or \$NX_ROOT\log on Windows.

Add the Notification Method (UNIX)

The following steps create a notification method shell script that sends the notification message to the service desk printer, SDPR2. In this example, the notification message will consist of the message header and the message text from the message template:

1. Set up the shell script to assemble the notification text and transmit it, as follows:

```
#!/bin/sh  
  
echo "  
TO: $NX_NTF_USERID  
SUBJECT: $NX_NTF_SUMMARY  
MESSAGE:  
$NX_NTF_MESSAGE" |lp -dSDPR2
```

2. Name the executable file sd_print, and place it in any directory used for common scripts at your site, such as /usr/local/netbin.
3. Make the shell script an executable file using chmod.
4. Choose Notification Methods from Notifications on the Administration Interface.
5. Choose New from the File menu.
6. Enter data in these fields:

Symbol

SDPR2

Description

Send backup notification to service desk printer SDPR2

Notification Method

/usr/local/netbin/sd_print

7. Click the Save button to save the new record. Then click Close Window to close the detail window.

Example Syntax

The following examples explain how to add PROPERTIES value to a notification.

Properties: Call Request
{@{call_req_id.properties.0.label} {@{call_req_id.properties.0.value}}

Properties: Change Order
{@{change_id.property_list.0.label} {@{change_id.property_list.0.value}}

Properties: Issue
{@{issue_id.property_list.0.label} {@{issue_id.property_list.0.value}}

Where the number is the numeric position of the property, beginning at 0.
The examples above refer to the first property in the ticket's list of properties.
If a specified property does not exist – say you refer to property 4 but a ticket has only 1 property – no text is written.

Chapter 3: Custom Reports

Unicenter Service Desk lets you customize existing reports or design your own reports. You can:

- Customize existing reports to contain exactly the information you need, such as additional fields.
- Produce a new report with any information available from the database in a format that is useful to you.
- Pass arguments of variable information into the report by including command line arguments. Arguments can be values or expressions, such as the current value of a field or an SQL WHERE clause expression.
- Generate reports at the command line, from a script file, or from a menu option.

To generate a custom report, you need to follow these basic steps, which are discussed in more detail in this chapter:

1. Design the report:
 - Decide what data you want to include in the report
 - Create a report template that contains SQL-like queries, expressions, and functions to manipulate data, and statements to format the data for the printed page.
2. Generate the report from:
 - The command line
 - A Unicenter Service Desk menu option
 - A script file

Note: If you have a third-party database system you can use its report generating tools to create reports with data from the Unicenter Service Desk database. Unicenter Service Desk provides several database views that simplify the process of creating customized reports using third-party database systems as described in the *Customize Crystal Reports* (see page 64) in this chapter. See the documentation for your database system for information about reporting on databases. For more information on database views, see the topic *Database Views* in the “Report Generation” chapter of the *Administrator Guide*.

How to Design Reports

To design a custom report, you should have a basic understanding of the following concepts:

- Writing SQL queries.
- Programming, especially in C.
- Creating special programs or script files that you may need to execute before you execute the report template program. For example, you may want to create a program that prompts the user to enter an argument, such as the conditions for a WHERE clause. These programming techniques are not described in this chapter.

Note: Before creating a custom report, be sure to check if the report you need is already provided. Unicenter Service Desk provides a wide variety of Crystal and Microsoft Access reports, as well as runtime versions of these products to let you run the reports. For more information on reports, see the chapter "Report Generation" in the *Administrator Guide*.

Selecting Information for the Report

To help you select data from the Unicenter Service Desk database for customized reports, refer the appendix "Data Element Dictionary" (see page 215) in this guide. It lists database tables, fields, descriptions, and other database information.

How to Create a Report Template

A report template is a file that, when executed by a Unicenter Service Desk report program, generates a report of a particular design. A report template contains variable expressions, functions, and statements that define how the data is fetched, calculated, and printed. For complete descriptions of the variable expressions, functions, and statements that make up a report template, refer the Reference section (see page 49) at the end of this chapter. An example (see page 44) at the end of this section describes a sample report template that illustrates how to pass arguments into a report.

To create a report template, create a file containing the following types of report statements:

Block statements

Used to define the Unicenter Service Desk database tables from which data will be fetched and the actions that are to be performed on the fetched data.

Layout statements

Used to define how the data variables and literal text display on the report output.

Each of these types of statements is described in more detail in this chapter.

Note: Store all your .rpt files in a new directory, \$NX_ROOT/site/mods/rpt (UNIX) or *installation-directory\site\mods\rpt* (Windows). This will preserve them when you upgrade to a new release of Unicenter Service Desk.

Block Statements

Block statements provide the report template with its framework. They define the data to be manipulated and control the execution of the report. Block statements begin with a name that must be unique throughout the report template. They then have the following two sections:

Data query section

This contains SQL SELECT, WHERE, and SORT clauses to define which data will be fetched from the database.

Output program section

This defines the actions that are to be performed on the fetched data. It contains variable declarations, functions, and other block statements, including nested statements, which can be used to create conditional reports. It can also contain layout statements, which format and print the data as ASCII text.

A simplified version of the syntax of a block statement that shows the relationship between the two sections follows:

```
BLOCK blockname ("SELECT clause", "WHERE clause")  
      SORT clause {output program statements}
```

The BLOCK (see page 55) in the Reference section discusses the detailed version of the syntax, along with a description of each clause and parameter.

Layout Statements in Report Templates

Layout statements define how variables and literal text will appear on the report output:

- You can use the PAGE HEADER and PAGE FOOTER statements to place information at the top and bottom of each report page.
 - You can nest HEADER, HEADER2, FOOTER, and PRINT statements within the braces section of the parent BLOCK statement to create titles and summary totals for the various *reporting sections* (parts of the report output).
- Note:** When nesting, be careful not to confuse the braces used in layout statements with the braces that encompass the nested statements within a parent BLOCK statement.
- You can include literal text to create labels and line drawing characters to enhance the appearance of the report.

The layout statements are as follows:

PAGE HEADER

This places information at the top of each report page. It is placed outside the BLOCK statement.

PAGE FOOTER

This places information at the bottom of each report page. It is placed outside the BLOCK statement.

HEADER

This places information at the top of each reporting section. It is placed inside the BLOCK statement.

HEADER2

This places continuation header information at the top of each succeeding page of a reporting section, if that reporting section extends over multiple pages. It is placed inside the BLOCK statement.

FOOTER

This places information at the bottom of each reporting section. It is placed inside the BLOCK statement.

PRINT

This places the data in a reporting section. It is placed inside the BLOCK statement.

You can also use the following predefined variables in layout statements:

- CT prints the current time
- CD prints the current date

- PG prints the page number

Data Fields

A data field is any variable in a layout statement that results in a piece of data when you generate the report. Use the following guidelines when placing fields in your report template:

- Enclose data fields in square brackets ([]).
- The field's square brackets define its print space on each line of output. This space is the number of characters delimited by the square brackets, including the brackets. If the output of a variable is longer than the print space, the output is truncated. To ensure that the field has enough print space, you can add trailing spaces between the variable name and the closing bracket. For example, these trailing spaces allow for contacts with long names:

[contact]

- For output that is less than one line, the field can be closed with a greater than right angle bracket (>). This extends the print space to the right margin. For example, the right angle bracket used in a HEADER statement allows the current date to print without being truncated:

[CD >

Note: When the field is more than one line and the variable is flagged as MULTILINE, the right angle bracket (>) acts exactly the same as the right square bracket (]). If the print statement for a MULTILINE variable is closed with the right angle bracket (>), characters wrap on white space to stay within the field defined by the left bracket ([) and the right angle bracket (>). Also, if the variable is not MULTILINE, the right angle bracket (>) causes all the data to be displayed on the current line regardless of its length.

- A field in a layout statement can refer to a previously defined variable or a column name.
- To reference a variable or column name in another block statement, use the following syntax:

blockname::column | variable-name

Literal Text

Literal text allows you to include supplementary information in your report. It will appear on the report output exactly as specified in the template. To include literal text in a layout statement, place it on any line after the opening brace ({) and before the closing brace (}). Do not enclose it in quotes or square brackets.

In this example, "ACME Company" and "Page: " are interpreted as literal text by the Unicenter Service Desk report program:

PAGE HEADER {

```
ACME Company           Page: [PG]
}

The FOOTER (see page 57), HEADER (see page 58), HEADER2 (see
page 59), PAGE FOOTER (see page 60), PAGE HEADER (see page 61), and
PRINT (see page 62) in the Reference section later in this chapter
discusses the layout statements.
```

Variable Expressions in Report Templates

Every value that you want to appear on the report output can be assigned to a variable. Variable expressions let you:

- Manipulate Unicenter Service Desk data
- Use functions to perform calculations on fetched values

The following example creates a variable named *desc* to reference the contents of the chg_desc field in the Change Order window. The MULTILINE flag allows the variable to print in its entirety over multiple lines:

```
desc = description MULTILINE;
```

The following example prints the description. The output will be as long as the length defined in the brackets. If you want a longer description to appear, increase the number of spaces in the brackets.

```
PRINT { [desc ] }
```

See Variable Expressions in the Reference section later in this chapter for more information on variable expressions.

Example: Report Template

The following Affected Contact Report template shows how to create a report template. It produces a report that lists open change orders with the same affected contact:

```
PAGE HEADER {
    As Of: [CD>
    [CT>
}
PAGE FOOTER {
    Page: [PG>
}
BLOCK chg ("SELECT \
    chg_ref_num, description, priority, \
    status, category, assignee \
    FROM Change_Request",
    "WHERE #Change_Request.status = 'OP' \
    AND #Change_Request.requestor = #ca_contact.id \
    AND #ca_contact.last_name = ? \
    AND #ca_contact.first_name = ? \
    AND #ca_contact.middle_name = ? ", $1, $2, $3)
{
    BLOCK st ("SELECT sym FROM Change_Status",
        "WHERE code = ? ", chg::status) {}
    BLOCK (strlen(category)) cat ("SELECT sym FROM Change_Category",
        "WHERE code = ? ", chg::category) {}

HEADER {
    OPEN CHANGE ORDERS WITH SAME REQUESTOR/FROM CONTACT
    CHANGE ORDER Summary      Pri  Status   Category      Assignee
}
HEADER2 {
    CHANGE ORDER Summary      Pri  Status   Category      Assignee
-----
}
    num = chg_ref_num;
    desc = description MULTILINE;
    pr = deref (priority);
    stat = st::sym;
    catgry = cat::sym;
    asgn = deref (assignee);

PRINT {
    [num      ] [desc           ][pr ] [stat   ] [catgry       ] [asgn ]
}
}
```

Page Header

The PAGE HEADER statement tells what to print on the top of each page of the report. CD and CT are predefined variables that give the current date and time. They will appear in the header on the top of each page. Each of these fields ends with an angle bracket, which allows the field to expand towards the right margin. Because “As Of:” is outside of a field and because it is on a line after the opening brace, it will appear as literal text on the report output.

```
PAGE HEADER {                                As Of: [CD>
                                                [CT>
}
```

Page Footer

PAGE FOOTER includes the page number with “Page: ” as literal text.

```
PAGE FOOTER {
                Page: [PG>
}
```

Note: Since PAGE HEADER and PAGE FOOTER statements produce global headers and footers, they are not included in a BLOCK statement.

Reporting Section

The main BLOCK statement, along with its nested statements, creates a reporting section. A reporting section is usually only part of the data in the report, but this report has only one reporting section. The unique name of this block is chg.

The SELECT clause selects the columns to be included in the data for the report FROM three tables, but only where conditions specified by the WHERE clause are met.

The last three AND expressions in the WHERE clause contain question marks, which act as argument placeholders that take the values of the \$1, \$2, and \$3 arguments, in order. Thus \$1 is for ca_contact.last_name, \$2 is for ca_contact.first_name, and \$3 is for ca_contact.middle_name. The \$1, \$2, and \$3 arguments obtain the values of command line arguments. For a command line example, see The Report Command topic in this chapter.

```
BLOCK chg ("SELECT \
...
"WHERE \
...
AND #ca_contact.last_name = ? \
AND #ca_contact.first_name = ? \
AND #ca_contact.middle_name = ? ", $1, $2, $3)
```

Reporting Section Headers

The opening brace starts the output program part of the BLOCK statement: its statements tell what to do with the data fetched by the SELECT and WHERE clauses. This example has nested HEADER and HEADER2 statements that will apply to this reporting section only. HEADER2 prints only if the report output is on multiple pages.

```
{  
    ...  
    HEADER {  
        OPEN CHANGE ORDERS WITH SAME REQUESTOR/FROM CONTACT  
        CHANGE ORDER Summary      Pri   Status Category Assignee  
    }  
    HEADER2 {  
        CHANGE ORDER Summary      Pri   Status Category Assignee  
        -----  
    }  
}
```

Variable Assignments

Here are the variable expressions that act on the data specified by the SELECT clauses. They assign variables to the values of columns and to the results of expressions. These variables match the fields in the PRINT statement that follows.

The MULTILINE flag on the *desc* variable causes them to print or display on multiple lines rather than being truncated. The deref function is used to return the string expression contained in the referenced columns.

```
num = chg_ref_num;  
desc = description MULTILINE;  
pr = deref (priority);  
stat = st::sym;  
catgry = cat::sym;  
asgn = deref (assignee);
```

Printing

The PRINT statement contains the fields to be printed. This statement could have also included literal text of lines that could enhance the appearance of the report. The final ending brace matches the opening brace of the output program section of the BLOCK statement.

```
PRINT {  
    [num ] [desc ] [pr] [stat] [catgry] [asgn ] }
```

```
}
```

```
}
```

How To Generate Reports

After you create the report template, you can generate the report by running the Unicenter Service Desk report program. The program can be executed from:

- The command line
- A Unicenter Service Desk menu option
- A script file

Note: If you are working on a UNIX server, you can include the report output redirector (rptuiDsp) parameter with the report command to display a dialog with options for sending the report to the screen, a file, or the printer. See Displaying a Dialog (UNIX Only) later in this chapter for details.

The Report Command

To generate a report from the command line, on UNIX, you must use the Unicenter Service Desk report command:

```
pdm_task report [-h][-e][-f][-F ffstring][-p pagelength] filename [command-line arguments]
```

Note: The report command is preceded by the pdm_task command, which sets necessary environment variables. If the report is designed to accept command line arguments, you must enter one for each argument in the report template.

On Windows, use the rpt_srv command:

```
rpt_srv report-title
```

For a complete description of each option in the report and rpt_srv commands, see the *CA Reference Guide*.

The following example includes the three command line arguments (Smith, Jane, and L) needed for the Affected Contact Report described in the report template example earlier in this chapter:

```
pdm_task report /reports/myrpt.rpt Smith Jane L
```

If an argument is empty, you must use a null string. For example, if Jane Smith did not have a middle initial, the syntax would be:

```
pdm_task report /reports/myrpt.rpt Smith Jane ""
rpt_srv \reports\myrpt.rpt Smith Jane L
```

How to Display a Dialog (UNIX Only)

You can include the report output redirector (rptuiDsp) parameter with the report command to display a dialog. The dialog displays the options to print the report to a file, display it in an Xterm window, or send it to the printer. For example:

```
pdm_task rptuiDsp report /reports/myrpt.rpt Smith Jane L
```

This example adds the title "Inventory Report" to the dialog:

```
pdm_task rptuiDsp report /reports/myrpt.rpt Smith Jane L "title:Inventory Report"
```

Report Template Reference

This section describes the variable expressions, functions, and statements in a report template.

Report Template Variable Expressions

Variable expressions define the data to be printed or displayed. They are placed in a layout or block statement.

Syntax

Syntax refers to the rules governing the formation of statements in a programming language. The structure of this string is as follows:

variable-name = *expression* [*flags*]

Flags

Flags format the result of a variable expression. Use these flags to format text fields:

MULTILINE

Displays on multiple lines rather than truncating.

RIGHT

Right justifies.

Use these flags to format numeric fields:

BLANKZERO

Functions as null-value fields, which do not print a zero.

BOOL

Converts zero to no or non-zero to yes.

REAL

Displays as floating point (default is integer).

ZEROFILL

Shows leading or trailing zeros

Use these flags to format date and time fields:

DATE

Shows only date portion of date/time.

DAYS

Displays durations with days.

HOURS

Displays durations with hours.

MINUTES

Displays durations with minutes.

SECONDS

Displays durations with seconds.

TIME

Shows only time portion of date/time.

Example

```
desc = description MULTILINE
```

Remarks

Variable names must be unique within a BLOCK statement and must not duplicate any column in the SELECT clause for the block. The same variable name can be used in different BLOCK statements but it cannot be repeated within a BLOCK statement.

Follow these syntax rules when including expressions in your report template:

- Use any valid C expression.
- Do not enclose variable or column names in quotes.
- Enclose string constants in single or double quotes.
- You can refer to a nested block, but only if it contains exactly one row.
- To include a column name that is the same as a keyword, precede the column name with a backslash (\). For example, ALIAS is a keyword and \alias is a column name.
- Use the dollar sign (\$) to reference environment variables, such as \$name, and to reference command line arguments, such as \$n, where n is the position of the argument on the command line.
- To specify the number of command line arguments, use \$#. For example, the following expression means that if the number of command line arguments is greater than one, use the additional argument as an argument; otherwise, set the value of the argument to an empty string. Note that the report template itself is considered a command line argument. Therefore, the number of arguments is at least one.

```
$# > 1 ? $1 : "
```

- Use ## to concatenate two strings, for example:

```
title = "This is the " ## "first line. "
long_name = fn
irst_name ## last_name
```

- The following casts are supported:
 - (number)
 - (string)
 - (date_time)
 - (duration)
- To reference a variable or column name in another block, precede the name with its block name and two colons. For example:

```
blockname::column | variable-name
```

Report Template Functions

The following functions can be used in your report template:

is_null (expr)

This function returns true if the expression is null.

```
false = 0  
true = is_null (false)
```

sqrt (expr)

This function calculates the square root of the expression.

```
nine = 9  
three = sqrt (nine)
```

pow (expr1, expr2)

This function raises *expr1* to the power *expr2*.

```
two = 2  
three = 3  
eight = pow (two,three)
```

log (expr, expr)

This function calculates the natural log of the expression.

```
ten = 10  
result = log (ten)
```

catname (expr, expr, expr)

This concatenates three strings representing a contact name into a string with commas, according to rules in the field format file.

```
last = "Murphy"  
first = "Fred"  
middle = "P"  
contact_name=catname (last, first, middle)
```

strlen (string)

This function returns the length of the string.

```
buffer = "A thirty character long string"  
thirty = strlen(buffer)
```

strindex (string, pattern [, start_index])

This function returns the index of the first pattern match, or the next pattern match after the *start_index*, in the string. Returns -1 if there is no match.

```
buffer = "A thirty character long string"
```

```
zero = strindex(buffer, " [A-Z] ")
```

```
two = strindex(buffer, " [a-z] ")
```

substr (*string, pattern [, length*)

This function returns the portion of the string after the first pattern match. If length is defined, it limits the length of the output string. Returns a string of zero length, if there is no match.

```
buffer = "A thirty character long string"
```

```
last_word = substr(buffer, " [a-z]*$ ")
```

```
first_capital_letter = substr(buffer, " [A-Z] ",
```

```
1)
```

substr (*string, index [, length*)

This function returns the portion of the string after the index. Its length is defined and limits the length of the output string. Returns a string of zero length, if there is no match.

```
buffer = "Summary: The network card displays a
```

```
code of ... "
```

```
summary = substr(buffer, 9)
```

```
30_char_summary = strindex(buffer, 9, 30)
```

The remaining functions (pseudofunctions) perform on a block of data rather than on variable expressions. These functions are usually placed in a BLOCK statement to get information about a nested BLOCK statement's data.

count (*block-name*)

This returns the number of rows in the block specified in the BLOCK statement. The block-name must be a simple string.

```
BLOCK sample ("SELECT id FROM Contact") {
```

```
entries = count (sample)
```

```
}
```

sum (*block-name, expr*)

This executes the expression for each row of the specified block and sums the result.

```
BLOCK sample ("SELECT actual_cost, est_cost FROM Change_Request") {
```

```
difference = sum (sample, est_cost-actual_cost)
```

```
}
```

average (*block-name, expr*)

This executes the expression for each row of the block and returns the average of the result.

```
BLOCK sample ("SELECT actual_cost, est_cost FROM Change_Request"){  
    avg_difference = average (sample, est_cost-actual_cost)  
}
```

prev (*expr*)

This returns the previous value of the expression. This function should be used with caution so its value does not overwrite the latest value by accident.

downtime (*sla_schedule*, *expr1*, *expr2* [, *delay-block*, *expr*, *expr*])

This invokes an SLA downtime calculation. The first argument must be a string that identifies a workshift. The other arguments are start and end times:

expr1 is the start date/time of the event

expr2 is the end date/time of the event

In this example, the wrkshft BLOCK fetches the work shift schedule, the evt_dly BLOCK statement fetches the delays and the downtime function uses these records to calculate the downtime.

```
BLOCK attevt ("SELECT start_time, fire_time, event_tmpl, obj_id FROM  
Attached_Events")  
{  
    BLOCK evt ("SELECT persid, sym, work_shift FROM Events ", "WHERE persid = ?",  
    attevt::event_tmpl) {}  
    BLOCK wrkshft ("SELECT sched FROM Bop_Workshift", "WHERE persid = ?",  
    evt::work_shift) {}  
  
    BLOCK evt_dly ("SELECT start_time, stop_time FROM Event_Delay", "WHERE obj_id  
= ?", attevt::obj_id) {}  
    total_downtime = downtime(wrkshft::sched,  
    attevt::start_time, attevt::fire_time,  
    evt_dly,  
    evt_dly::start_time, evt_dly::stop_time);  
}
```

deref (*column-name*)

This returns the string representation of the pointer by performing an automatic lookup in the appropriate table.

```
BLOCK chg ("SELECT organization FROM Change_Request") {  
    org = deref (organization)
```

```
}
```

Because this pseudofunction involves lookups, it is valid only if it is the only thing in the expression. For example, this is valid:

```
model = deref (nr_model)
```

This is not valid:

```
model = "model" ## deref (nr_model)
```

Note: Forward references to variables or blocks are not allowed.

Report Template BLOCK Statements

Block statements define the database tables from which data will be fetched. Can include actions to perform on the fetched data.

Syntax

Syntax refers to the rules governing the formation of statements in a programming language. The structure of this string for BLOCK is as follows:

```
BLOCK blockname (
    "SELECT [ALIAS,] field_name[, field_name ...]
     FROM table_name[, table_name ...] "
     [, "WHERE where_clause"[, arguments,,] ]
     [SORT "sort clause"]
{
    output program statements
}
```

Parameters

blockname

Identifies the block. Each *blockname* must be unique.

SELECT clause

Follows the *blockname* and is delimited by double quotes. Lists the columns to be fetched, followed by the keyword FROM, followed by the tables from which the columns are to be fetched. It is required. Here is an example with three tables specified:

```
"SELECT open_date, chg_ref_num \
last_name, first_name \
FROM Change_Request, \
ca_contact"
```

You cannot include an SQL alias, such as:

```
"SELECT open_date As OpenDate"
```

WHERE clause

Optionally follows the SELECT clause and further qualifies the information selected. It may be a string constant or an expression evaluating to a string. If the WHERE clause is an empty string, all records are returned. WHERE clauses can contain replacement arguments (which refer to variables or command line arguments) using the syntax of a question mark (?). The following WHERE clause could follow the previous SELECT clause:

```
"WHERE #Change_Request.open_date >= ? \
AND #Change_Request.active_flag = 1 \
AND #ca_contact.last_name = ?, $1"
```

Note: The WHERE clause must be separate from the SELECT clause because the WHERE clause can be an expression evaluating to a string, whereas the SELECT clause is exclusively a string constant. This gives you more flexibility and data manipulation capabilities in producing your report.

SORT clause

Optionally follows the SELECT and WHERE clauses and sorts the fetched rows of data. The SORT clause is formatted like the SQL ORDER BY clause. Here is an example:

```
SORT "open_date"
```

Output program statements

Control execution of the report. Before the data query, the HEADER statement, if included, prints the header test for the block. The data query then runs. If data is returned, each statement executes in the order written, with one exception. Block functions, like sum and average, behave as though they were at the end of the output program. In fact, their values are not stable until execution begins on the next data record.

It is important to remember that the output program depends on the success of the data query. If no data is returned from the query, then, except for the HEADER statement, the output program will not execute.

Example

This BLOCK statement assumes that an argument will be passed that holds an integer equal to the change order priority. The WHERE clause first checks the number of arguments passed (\$#). If one is present, it is used to evaluate the expression to produce the WHERE clause; otherwise a null WHERE clause is substituted ("").

```
BLOCK chg ("SELECT priority FROM Change_Request",
$# > 1 ? "WHERE priority =## $1 : "") {}
```

Example

```
FOOTER {
    Summary Information:
        Total Failures: [Fail_count >
        Total Downtime: [Downtime >
    }
```

Remarks

HEADER, HEADER2, FOOTER, PRINT and variable expressions can be placed in the braces. Any statement will be executed for each row selected.

Note: PAGE HEADER and PAGE FOOTER statements cannot be placed in a BLOCK statement.

Report Template FOOTER Statements

This layout statement places information at the bottom of a reporting section.

Syntax

Syntax refers to the rules governing the formation of statements in a programming language. The structure of this string for FOOTER is as follows:

FOOTER {parameters}

Parameters

The parameters are as follows:

CD

A predefined variable used to display the current date.

CT

A predefined variable used to display the current time.

PG

A predefined variable used to display the current page number.

column | variable-name

This field can be a variable from an earlier variable expression or a reference to a column in the SQL clause of a BLOCK statement.

literal-text

Any text that is not a predefined variable or a column or variable name is interpreted as literal text. Literal text that you include in the FOOTER statement appears in the exact horizontal location where you enter it.

Remarks

FOOTER statements are printed at the bottom of a reporting section. A typical use might be to present summary information or statistics. You can include a FOOTER statement in a BLOCK statement.

A field's content occupies the exact space delineated by the square brackets. Any excess characters are truncated. However, you can close a field with an angle bracket (>) to permit its content to expand in its entirety towards the right margin.

To reference a variable or column name in another BLOCK statement, use the following syntax:

blockname::column | variable-name

Report Template HEADER Statements

This layout statement places information at the top of a reporting section.

Syntax

Syntax refers to the rules governing the formation of statements in a programming language. The structure of this string for HEADER is as follows:

```
HEADER {parameters}
```

Parameters

See Header for a list and explanation of the valid parameters for this statement.

Example

```
HEADER {
    Contact Summary Report
    Contact Name  Contact Alias  Organization
}
```

Remarks

HEADERS are printed at the beginning of a reporting section and can be included in a BLOCK statement. HEADERS are typically used to present section and/or column headings.

A field's content occupies the exact space delineated by the square brackets. Any excess characters are truncated. However, you can close a field with an angle bracket (>) to permit its content to expand in its entirety towards the right margin.

Note: If the print statement for a MULTILINE variable is closed with the right angle bracket (>), characters wrap on white space to stay within the field defined by the left bracket ([) and the right angle bracket (>). Also, if the variable is not MULTILINE, the right angle bracket (>) causes all the data to be displayed on the current line regardless of its length.

To reference a variable or column name in another BLOCK statement, use the following syntax:

```
blockname::column | variable-name
```

Report Template HEADER2 Statements

This layout statement places continuation HEADER information at the top of each succeeding page of a reporting section, if that reporting section extends over multiple pages.

Syntax

Syntax refers to the rules governing the formation of statements in a programming language. The structure of this string for HEADER2 is as follows:

```
HEADER2 {parameters}
```

Parameters

See FOOTER for a list and explanation of the valid parameters for this statement.

Example

```
HEADER2 {
    Contact Summary Report (continued)
    Contact Name   Contact Alias   Organization
}
```

Remarks

A HEADER2 statement can be included in a BLOCK statement.

A field's content occupies the exact space delineated by the square brackets. Any excess characters are truncated. However, you can close a field with an angle bracket (>) to permit its content to expand in its entirety towards the right margin.

To reference a variable or column name in another BLOCK statement, use the following syntax:

```
blockname::column | variable-name
```

Report Template PAGE FOOTER Statements

This layout statement places information at the bottom of each report page.

Syntax

```
PAGE FOOTER {parameters}
```

Parameters

With the exception that you cannot use column and variable names, the parameters for this statement are the same as those for FOOTER. See FOOTER for a list and explanation of the valid parameters for this statement.

Example

```
PAGE FOOTER {  
    Page Number: [PG>  
}
```

Remarks

A field's content occupies the exact space delineated by the square brackets. Any excess characters are truncated. However, you can close a field with an angle bracket (>) to permit its content to expand in its entirety towards the right margin.

To reference a variable or column name in another BLOCK statement, use the following syntax:

blockname::column | variable-name

Report Template PAGE HEADER Statements

This layout statement places information at the top of each report page.

Syntax

Syntax refers to the rules governing the formation of statements in a programming language. The structure of this string for PAGE HEADER is as follows:

PAGE HEADER *{parameters}*

Parameters

With the exception that you cannot use column and variable names, the parameters for this statement are the same as those for FOOTER. See FOOTER for a list and explanation of the valid parameters for this statement.

Example

```
PAGE HEADER {  
    Date of Report: [CD>  
    Time of Report: [CT>  
}
```

Remarks

PAGE HEADERS are printed at the top of every report page. They can be defined at any point within the report template file, but they cannot be included within a BLOCK statement.

A field's content occupies the exact space delineated by the square brackets. Any excess characters are truncated. However, you can close a field with an angle bracket (>) to permit its content to expand in its entirety towards the right margin.

To reference a variable or column name in another BLOCK statement, use the following syntax:

blockname::column | variable-name

Report Template PRINT Statements

This layout statement places data in a reporting section.

Syntax

Syntax refers to the rules governing the formation of statements in a programming language. The structure of this string for PRINT is as follows:

PRINT {parameters}

Parameters

Refer FOOTER (see page 57) for a list and explanation of the valid parameters for this statement.

Example

```
PRINT {
  [num ] [desc ] [pr] [stat] [catgry] [asgn ]
}
```

Remarks

Place PRINT where you want the data for a reporting section to appear in the report. You can include a PRINT statement in a BLOCK statement.

A field's content occupies the exact space delineated by the square brackets. Any excess characters are truncated. However, you can close a field with an angle bracket (>) to permit its content to expand in its entirety towards the right margin.

Note: If the print statement for a MULTILINE variable is closed with the right angle bracket (>), characters wrap on white space to stay within the field defined by the left bracket ([) and the right angle bracket (>). Also, if the variable is not MULTILINE, the right angle bracket (>) causes all the data to be displayed on the current line regardless of its length.

To reference a variable or column name in another BLOCK statement, use the following syntax:

blockname::column | variable-name

Customize Crystal Reports

Before you can display any of these reports, the following conditions apply:

- You must make the Crystal reports available to the Crystal Report Selector by copying them to the Crystal directory: \$NX_ROOT/bopcfg/rpt.
- Your database client must be up and running, with connectivity established to the database server running on the same or another machine. If you are using a Service Desk Client to run your Crystal or Access reports you need to have installed a database client for the specific database (such as Ingres) and have established connectivity with the database server in order to run these reports.

After creating any custom Crystal reports, perform the following:

1. Copy custom Crystal reports to the following crystal directory:
\$NX_ROOT/bopcfg/rpt
2. Add the file names of custom Crystal reports to the following configuration file:
crystal.cfg

You can then access the Crystal reports by clicking Start on the taskbar, and then choosing Reporting, Service Desk Reporting (Crystal Reports) from the Unicenter Service Desk menu (accessible from the Programs menu). The Service Desk Reporting (Crystal) window appears (see the procedure Displaying Crystal Reports in the online help for more information).

Important! Unicenter Service Desk clients cannot be migrated. Therefore, if you create and use Crystal reports on the Unicenter Service Desk Server and you plan to upgrade your version of Service Desk, you need to copy all custom reports to a different location so you will not lose them. Following the migration, copy the reports back to the \$NX_ROOT/bopcfg/rpt Crystal directory and modify the crystal.config file to make them accessible from the Report Selector.

Chapter 4: Customization of Queries and Messages

Unicenter Service Desk provides a number of features that let you narrow the focus of information so you can concentrate on requests, change orders, and issues that apply to your immediate situation. One of these functions stores queries that you can use to see relevant information on the scoreboard of the Unicenter Service Desk administrative client or web interface. Another lets you customize the messages that notify key personnel of ticket activities.

The first part of this chapter explains how you can provide stored queries that focus on tickets related to the logged-in user and customize the counter fields in the scoreboard area of the administrative client and web interfaces. The second part explains how you can customize activity notification messages to include attributes from the activity log object and information on specific tickets.

Scoreboard Queries

One of the tables in the database, Cr_Stored_Queries, defines stored queries. These stored queries, which are very similar to SQL queries, can be used to customize the counter fields on nodes in the scoreboard area of the administrative client and web interfaces. The counter fields tell how many records match the query. For example, they can tell how many of various types of requests have been assigned to the logged-in user.

Each user can customize the counter fields that appear on his or her scoreboard (this is explained in the online help.) However, the system administrator must first define the various types of requests that can be counted in these counter fields as stored queries. For more information about scoreboard queries, see Setting Up Stored Queries in the chapter "Establish the Support Structure" of the *Administrator Guide* for instructions.

Note: Scoreboard counts will be incorrect if database query values are equal to NULL. For example, if your Scoreboard query specifies that assignee.organization != xyz, and an assignee field is blank (NULL) for a record, then that record will not be part of the Scoreboard count.

Stored Queries for Logged in User

Two of the fields that must be defined on the Stored Query Detail window are Where Clause and Label. Both of these fields can contain expressions that are customized to the logged-in user. Stored queries refer to objects and attributes (see page 461), rather than to table names and columns. A stored query that is customized to the logged-in user consists of two parts, as follows:

The object (such as cr for a request)

This is usually specified on the left of the equal (=) sign. The syntax for this part of the stored query is:

att_name[.att_name...].SREL_att_name

A stored query always has a Type, which is an object name that the query is executed against and provides context for the query. In the syntax above, the first att_name must be an attribute name of the context object.

The logged-in user (the instance of the cnt object for this user)

This must be specified on the right of the equal (=) sign if the tickets are to be selected based on an attribute of the logged-in user. The syntax for this part of the stored query is:

@att_name[.att_name...].SREL_att_name

Syntax for cr Object

Use this syntax if the reference is to the request (cr) object:

att_name[.att_name...].SREL_att_name

This example identifies the location of the person assigned to handle a ticket. In this example, the object name is omitted, as the type of the Stored Query implies the cr object:

`assignee.location=@cnt.location AND active=1`

where:

assignee

The attribute in the request object that maps to the assignee field in the corresponding table. For example, the assignee attribute is defined in the cr object with SREL agt, which means it refers to the agt factory. The agt factory is part of the cnt object definition.

location

The attribute in the cnt object that maps to the c_l_id field in the Contact table. The location attribute is defined in the cnt object with SREL loc, which means it refers to the loc object.

WHERE Clause

The following example demonstrates a value you can code in a WHERE clause:

`assignee.location=@cnt.location AND active=1`

Given the Stored Queries type is a Request, this query selects all active requests where the assignee's location is the same as the location of the logged-in user.

Label

Attributes in the cnt object can be included in labels the same way they are included in WHERE clauses. Here is an example of the use of an attribute in the cnt object in a label:

`@cnt.location.name Calls`

This label will include the name of a location, for example, Phoenix, where Phoenix is substituted for @cnt.location.name when the label is displayed on a window. The label will be displayed as Phoenix Calls.

The IN Keyword

The IN keyword allows a stored query to reference two (or more) tables without creating a join. This can result in significant efficiencies in executing the query. It is coded as follows:

```
SREL_att_name IN ( value1 [, value2 [...] ] )
```

For example, a request query could be coded as:

```
category.sym IN (\'Soft%\', \'Email\')
```

This results in the following SQL WHERE clause:

```
category IN (SELECT persid FROM prob_ctg WHERE sym LIKE 'Soft%' OR sym = 'Email')
```

One use of IN is to avoid Cartesian products. For example, the following query results in a Cartesian product and is very inefficient:

```
assignee.last_name LIKE 'MIS%' OR group.last_name LIKE 'MIS%'
```

By using IN, the query does not create a Cartesian product; in fact, it creates no joins at all, as illustrated by the following example:

```
assignee.last_name IN 'MIS%' OR group.last_name IN 'MIS%'
```

Note: The parentheses that normally enclose the list of values on the right side of IN can be omitted if there is only one value in the list. Similarly, you should avoid joins in data partitions by converting a data partition, illustrated as follows:

```
assignee.last_name LIKE 'Smith'
```

to:

```
assignee = U'374683AA82ACE34AB999A042F3A0BA2E'
```

where:

U

indicates that the value is a uuid.

'374683AA82ACE34AB999A042F3A0BA2E'

The 32 characters in single quotes indicates the string representation of an actual uuid.

This avoids the join with some loss in clarity. Using IN, the same partition can be written as illustrated in the next example, with the clarity of the first version and almost the same efficiency as the second version:

```
assignee.last_name IN 'Smith'
```

Service Desk supports the IN clause applied to QREL or BREL lists. For example, if you want to find all the Requests with Assets that are parents of another specific Asset (with id 374683AA82ACE34AB999A042F3A0BA2E), the appropriate where clause is as follows:

```
affected_resource.[parent]child_hier.child IN
(U'374683AA82ACE34AB999A042F3A0BA2E')
```

The first part of the clause, *affected_resource*, is an SREL (foreign key) of the cr (Request) object, pointing to the Network_Resource table. The *child_hier* portion is a list of hier objects pointing to the hierarchical relationships. The last part, *child*, forms the first part of the where clause for the IN sub query. The 374683AA82ACE34AB999A042F3A0BA2E portion is the foreign key value to match on *child*. *[parent]* specifies the sub query return. Since the id value is a string representation of a UUID it must be indicated as such and written as U'374683AA82ACE34AB999A042F3A0BA2E'

The following is an example of the actual SQL generated, which provides all the Requests where the Asset is a parent of a specific Asset:

```
SELECT Call_Req.id FROM Call_Req WHERE Call_Req.affected_rc IN (SELECT
hier_parent FROM Asset_Assignment WHERE hier_child =
U'374683AA82ACE34AB999A042F3A0BA2E')
```

To query on multiple parents, you can provide a comma-separated list in the () portion of the SQL, as shown by the following example:

```
affected_resource.[parent]child_hier.child IN
(U'374683AA82ACE34AB999A042F3A0BA2E', U'374683AA82ACE34AB999A042F3A0BA2E')
```

The attribute name in brackets ([]) is used to form the SELECT portion of the sub-clause. Bracket notation is not used for the group Stored Queries shipped with Service Desk 6.0, as illustrated in this example:

```
(assignee = @cnt.id OR group.group_list.member IN (@cnt.id)) AND active = 1
```

Note: If bracket notation is not used, the SQL subsystem assumes that it is the attribute name of the first symbol in the dot-notation portion. It works in this case, more out of luck, that the group_list object has an attribute named 'group' in it. If it were named anything else, the where clause would fail to parse! The equivalent clause with brackets illustrated as follows:

```
(assignee = @cnt.id OR group.[group]group_list.member IN (@cnt.id)) AND active =
1
```

Note: You cannot extend the dot notation. For example, the following does not work:

```
affected_resource.[parent]child_hier.child.name IN ('chicago1')
```

Time-Based Queries

Time spans can be used to create time-based stored queries. A time span specifies a period of time, which can be relative to the current date. For example, a time span could refer to today, yesterday, last week, or last month. A time span has a name, such as TODAY or YESTERDAY. You refer to a time span in a stored query by using either of two built-in functions, as follows:

StartAtTime (*timespan-name*)

This refers to the beginning of the period described by the time span.

EndAtTime (*timespan-name*)

This refers to the end of the period described by the time span.

The syntax rules for stored queries require that the time span name be enclosed in single quotes, with each single quote preceded by a backslash. For example, to refer to the beginning of last week, you would specify:

```
StartAtTime(\PAST_WEEK\')
```

The passage of time makes it necessary to periodically refresh a stored query containing a reference to a time span. For example, the interval described by "yesterday" changes at midnight. You specify the Start Time, End Time, and Trigger Time for refreshes in the Timespan Detail window.

Start Time

Start Time specifies the beginning of the time span in absolute or relative terms. The following table describes the fields within the Start Time section of the Timespan Detail window:

Year

An explicit year, such as 2000, or a relative year such as +1 (next year) or -1 (last year)

Month

An explicit month from 1 (January) to 12 (December), or a relative month such as +1 (next month) or -1 (last month)

Day

An explicit day from 1 to 31, or a relative day such as +1 (tomorrow) or -1 (yesterday)

Hour

An explicit hour from 0 to 24, or a relative hour such as +1 (next hour) or -1 (last hour)

Minute

An explicit minute from 0 to 59, or a relative minute such as +1 or -1

End Time

End Time specifies the end of the time span in absolute or relative terms. The End Time fields of the Timespan Detail window are the same as the Start Time (see page 71) fields of the Timespan Detail window.

Trigger Time

The Trigger Time field specifies when the WHERE clause of a stored query containing a reference to the time span is recreated and the stored query refreshed. Trigger Time must be relative to the current time as described in the following table:

Year

Must a relative year from -1 (last year) to +36 (36 years from now).

Month

Must a relative month from -1 (last month) to +11 (11 months from now).

Day

Must a relative day from -1 (yesterday) to +31 (31 days from now).

Hour

Must a relative hour from -1 (last hour) to +23 (23 hours from now).

Minute

Must a relative minutes from +9 (9 minutes from now) to +59 (59 minutes from now).

Query based on Priority

In the database, the Priority table has two columns namely sym and enum. The value the users see are the sym values. But the application sees the sym based on the enum values. At present, the default sym values 1 to 5 are reversed in their enum value.

Example:

Sym	Enum
1	5
2	4
3	3
4	2
5	1

Thus when writing the stored query, when you reference a value of 5, you are actually looking for priority 1 unless you use a .sym to specify which attribute to look at.

Important! You should never play with the default enum values assigned. Instead, if adding new sym values, just continue from the highest enum value and so on.

ITIL-Specific Queries

Problems and Incidents are simply requests with a certain value in the "type" attribute, "I" for Incidents and "P" for Problems.

The following stored query will list all Incidents that the Assignee's Organization or the Group's Organization equals the logged in Analysts Organization:

```
assignee.organization IN @cnt.organization OR group.organization IN  
@cnt.organization) AND active = 1 AND type = '\'I\''
```

For Problems the query is identical except the type = '\'P\''

Activity Notification Messages Customization

Notification messages can be sent automatically when request activities occur. For information about notification messages see Setting Up Activity Notifications in the chapter "Policy Implementation" of the *Administrator Guide*. That section contains instructions for defining activity notifications using Request Management.

Two of the fields that must be defined on the Activity Notifications Detail window are Notification Message Title and Notification Message Body. Both of these fields can contain attributes from the activity log object (alg for Requests/Incidents/Problems, chgalg for Change Orders and issalg for Issues. These three activity log objects are almost identical) and can identify the specific request related to the activity.

Formatting Attributes for Activity Notifications

Optional formatting and escaping of individual attributes can be achieved using the properties listed below. This can be useful especially if formatting HTML notification where the data in the attribute may need to be escaped to conform to HTML standards.

To include formatting, use the following syntax:

```
@{property=value property=value:attribute_name}
```

Property values pairs are separated by at least one space and are not case sensitive. A colon separates the formatting properties from the attribute name. If no properties are listed, no formatting or escaping will be done on attribute.

The following lists the available formatting properties:

Property	Description
DATE_FMT	Specifies the date format for attribute. Valid values are: MM/DD/YYYY MM-DD-YYYY DD/MM/YYYY DD-MM-YYYY YYYY/MM/DD YYYY-MM-DD Valid only for Date attributes. Dates embedded in strings are not affected.

Property	Description
ESC_STYLE=NONE HTML URL	<p>Specifies the escape type of the formatted text. Valid values are:</p> <p>NONE</p> <p>Default setting. Specifies that no special treatment be given to any character in the content body.</p> <p>HTML</p> <p>Give special treatment to the following characters, which are meaningful in HTML text:</p> <p>& becomes &#38; " becomes &quot; < becomes &lt; > becomes %gt;</p> <p>URL</p> <p>Translate all characters other than letters, digits, and '@*-_.#' to '%xx', where xx is the hexadecimal coding of the translated character.</p>

Property	Description
JUSTIFY=LEFT CENTER RIGHT TRUNCATE WRAP LINE	<p>Specifies the justification of the formatted text. Valid values include:</p> <p>TRUNCATE</p> <p>(default if formatting) Truncates text to WIDTH property value if a positive integer. If ESC_STYLE=HTML, eliminates HTML formatting by replacing '<' and '>' with &lt; and &gt; (but see KEEPLINKS and KEEPTAGS).</p> <p>LEFT CENTER RIGHT</p> <p>Produces exactly WIDTH characters, truncated or padded with spaces as necessary, with any embedded new lines replaced by a single space. If ESC_STYLE=HTML, the output text is delimited by <pre> and </pre> tags. The WIDTH argument must be specified as a positive integer.</p>
	<p>WRAP</p>
	<p>Same as LEFT, except that text wrapping honors word boundaries (line breaks are not placed within words).</p>
	<p>LINE</p>
	<p>Same as TRUNCATE, except that it also replaces all embedded line breaks with
 tags if</p>
	<p>ESC_STYLE=HTML.</p>
KEEPLINKS=YES NO	<p>If KEEPLINKS=YES is specified, the action of JUSTIFY=LINE or JUSTIFY=TRUNCATE is modified to preserve HTML anchor tags (Action:) while</p>
	<p>converting all other '<' and '>' characters. Mutually exclusive with KEEPTAGS. Only valid if</p>
	<p>ESC_STYLE=HTML.</p>
KEEPNL=YES NO	<p>The normal action of PDM_FMT is to convert all embedded newlines and any following spaces to a single space. If KEEPNL=YES is specified, embedded newlines are preserved. This argument is ignored for JUSTIFY=LINE.</p>
KEEPTAGS=YES NO	<p>If KEEPTAGS=YES is specified, the action of JUSTIFY=LINE or JUSTIFY=TRUNCATE is modified to preserve all HTML tags. Mutually exclusive with</p>
	<p>KEEPLINKS. Only valid if ESC_STYLE=HTML.</p>

Property	Description
PAD=YES NO	If PAD=NO is specified, PDM_FMT does not convert empty strings to a single space. This is the normal action when WIDTH is non-zero, or JUSTIFY is TRUNCATE or WRAP.
WIDTH= <i>nn</i>	When non-zero, specifies that the text should be formatted to exactly WIDTH characters.

For example, to format the Request description for an HTML notification by escaping HTML specific characters, adding
 tags for line breaks and keeping any HTML Links as links, enter the following:

```
@{ESC_STYLE=HTML JUSTIFY=LINE KEEPLINKS=YES:call_req_id.description}
```

To format the open_date of a Request to European format, enter the following:

```
@{DATE_FMT=DD-MM-YYYY:call_req_id.open_date}
```

Attributes from the Activity Log Object

To include an attribute from the activity log object, include this in the Notification Message Title or Notification Message Body field:

```
@{att_name}
```

Note that the name of the object, alg or chgalg or issalg, is the default and need not be specified. For example, to include the type of activity in the message title, enter this in the Notification Message Title field (along with the rest of what you want in the title):

```
@{type}
```

To include the description of the activity in the message body, enter this in the Notification Message Body field (along with the rest of what you want in the body):

```
@{description}
```

Information on Specific Requests

For messages to provide information on the specific request that triggered the notification, the Notification Message Title or Notification Message Body fields must contain an attribute in the activity log object that references the request object. Enter this reference in this format:

`@{call_req_id.cr_att_name}`

where:

`@`

Indicates to replace this expression.

`call_req_id`

The attribute in the activity log object that links it to a specific instantiation of the request object (cr).

`cr_att_name`

Any attribute in the cr object.

For example, to include the impact of the request in the message title, enter this in the Notification Message Title field (along with the rest of what you want in the title):

`@{call_req_id.impact.sym}`

To identify the affected resource in the message body, enter this in the Notification Message Body field (along with the rest of what you want in the body):

`@{call_req_id.affected_resource.name}`

To specify to reopen a specific request by number, enter this in the Notification Message Title field:

Reopen Request `@{call_req_id.ref_num}`

There are several other mechanisms by which messages can be sent which are in the context of the request itself (or change order or issue). When the context is the request itself, you do not need (and cannot use) the "call_req_id" part of the reference. So, in these cases, you need to use:

`"@{ref_num}" rather than "@{call_req_id.ref_num}"`

Information on Specific Change Orders

For messages to provide information on the specific change order that triggered the notification, the Notification Message Title or Notification Message Body fields must contain an attribute in the activity log object that references the change order object. Enter this reference in this format:

`@{change_id.chg_att_name}`

where:

`@`

Indicates to replace this expression.

change_id

The attribute in the activity log object that links it to a specific instantiation of the change order object (chg).

chg_att_name

Any attribute in the chg object.

For example, to include the priority of the change order or in the message title, enter this in the Notification Message Title field (along with the rest of what you want in the title):

`@{change_id.priority.sym}`

To identify who reported the change order (Affected End User) in the message body, enter this in the Notification Message Body field (along with the rest of what you want in the body):

`@{change_id.requestor.combo_name}`

To specify to reopen a specific change order by number, enter this in the Notification Message Title field:

Reopen Change Order `@{change_id.chg_ref_num}`

Note: For messages to provide information on an issue that triggered a notification, the Notification Message Title or Notification Message Body fields must contain an attribute in the activity log object that references the issue object, iss. Using the information for requests and change orders presented here along with the details in the appendix "Objects and At (see page 461)tributes" you can easily see how to accomplish this.

For example, to include the priority of the issue or in the message title, enter the following in the Notification Message Title field (along with the rest of what you want in the title):

`@{issue_id.priority.sym}`

Chapter 5: Customizing the Schema Using the Web Screen Painter

You can use Web Screen Painter (WSP) Schema Designer to modify the flexible database schema of Unicenter Service Desk to meet your needs. The Schema Designer provides an easy-to-use graphical user interface to review and modify the Unicenter Service Desk schema. WSP also allows you to test your schema changes on your own web forms before updating the physical DBMS schema or affecting other users.

Here are the kinds of schema changes you can make and use in your own forms and reports:

- Add new tables to the database
- Add new columns to existing tables
- Make a column required
- Change a table or column's display name or function group

Notes

1. You cannot use WSP to change the length of an existing column, and it is strongly recommended that you not use other tools to do so. Changes to the length of an existing column are not supported, and may cause other applications accessing the Unicenter Service Desk database to fail.

Important! Never under any circumstances shorten a field or delete an existing field, as this could cause Unicenter Service Desk itself to fail.

2. Be careful when adding columns to an existing table, as you can inadvertently exceed the record length capacity of the underlying database. Check the specifications for the database that you are using with Unicenter Service Desk and make modifications within the limits of that database..
3. Publishing changes to the database schema could require limited or considerable downtime, depending on the changes you make and the capabilities of your underlying database.
4. If you are a new user of Unicenter Service Desk, it is easier to make all of your changes during testing instead of waiting until you are in production.

The beginning of this chapter reviews general procedures you must perform before and after changing the database schema. The rest of the chapter contains specific procedures you can use to customize your schema. Most of these procedures are followed by an example of a change you might want to make to the standard database schema.

Important! WSP verifies that the first letter of a new table or column name is "z", and inserts a "z" if necessary. This guarantees that your user-defined field names do not conflict with field names used by Unicenter Service Desk, now or in future releases.

Schema Designer Modification Overview

Follow these steps to modify the Unicenter Service Desk schema:

1. Make your changes with the WSP Schema Designer. Changes can include modifying existing tables and columns and defining new ones.
2. Put your changes into test mode. Changes in test mode are defined to the Object Engine associated with WSP, but are not defined to the physical database. WSP users can access the modified schema, but normal Service Desk users are not exposed to them.
3. Update or create web forms making use of the modified schema. You can examine data in your web forms, and even create or update records in site-defined tables without affecting the Unicenter Service Desk database. All updates affect only the Object Engine associated with WSP.
4. Repeat steps 1 through 3 until you are satisfied with your schema changes and with the web forms that use them.
5. Publish your schema changes. Publishing requires taking down Unicenter Service Desk.

We recommend you use the Unicenter Service Desk Change Manager to schedule and get approval for publishing schema, which is a two-step process:

1. Use WSP to create or update files on all Unicenter Service Desk servers describing the modified schema.
2. Run the pdm_publish script on the primary service to modify the physical DBMS. You must shutdown Unicenter Service Desk services prior to running the pdm_publish script.

The WSP Schema Designer Tool

You can make changes to Unicenter Service Desk using the Schema Designer tool of Web Screen Painter. To show this tool:

1. Start WSP by one of the following ways:
 - a. From the Windows Start menu, select Program Files, CA, Unicenter Service Desk, Web Screen Painter.
 - b. From the Windows Start menu, select Run and in the command line window, enter the command pdm_wsp.
 - c. In Linux, you can start Web Screen Painter by entering the command pdm_wsp with \$NX_ROOT/bin in your path.

A WSP login window appears.

2. Enter your User Name and Password.
3. Choose Schema Designer from the Tools menu.

The Schema Designer window appears:

The left side of the Schema Designer window shows the Unicenter Service Desk database in a tree format. The initial display lists tables, each preceded by a plus sign and a yellow folder icon. To see the columns in a table, either double-click the table name, or click on the plus sign. When you do this, the plus sign changes to a minus sign, and WSP displays the columns in the table in a tree format.

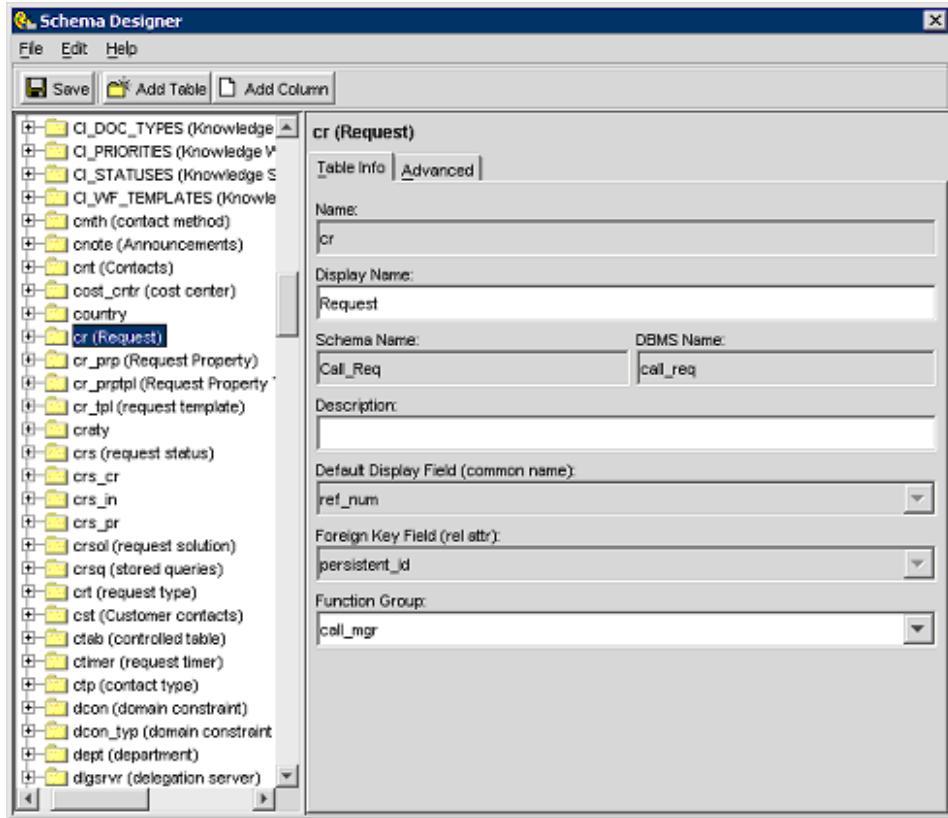
WSP shows both tables and columns in sequence by their Object Name. In addition, if the Display Name differs from the Object Name of the table or column, WSP shows the Display Name in parentheses after the Object Name.

The right side of the window displays the properties of the table or the column that is selected.

Schema Designer Tabs

Table Info Tab

When you click on a table from the Tables tree, WSP populates the Table Info tab with the information pertaining to the table.



The following information is shown in the Table Info tab:

Name

The object name of the table. For example, the object name of the cr table is "cr". This is a read-only field.

Display Name

The user-friendly name of the table. For example, the Display Name of the cr table is "Request". You can change the Display Name of a table by entering a new name in this field.

Schema Name

The name used to refer to the table in Unicenter Service Desk utilities, such as pdm_userload. This field is read-only for standard tables. For site-defined tables, Schema Name defaults to the Object Name. You can change the Schema Name by entering a new value in this field.

DBMS Name

The name used to refer to the table in the physical DBMS. This field is read-only for all tables. For site-defined tables, it is always the same as Schema Name.

Description

A brief description of the table.

Default Display Field (common name)

The column displayed on the UI for a field that references this table. For example, the assignee field of a request is a reference to the Contact table. Since the common name of the Contact table is combo_name (last, first middle), the combo name of the referenced contact is shown on the UI as the value of assignee. You cannot change the value of common name.

Foreign Key Field (rel attr)

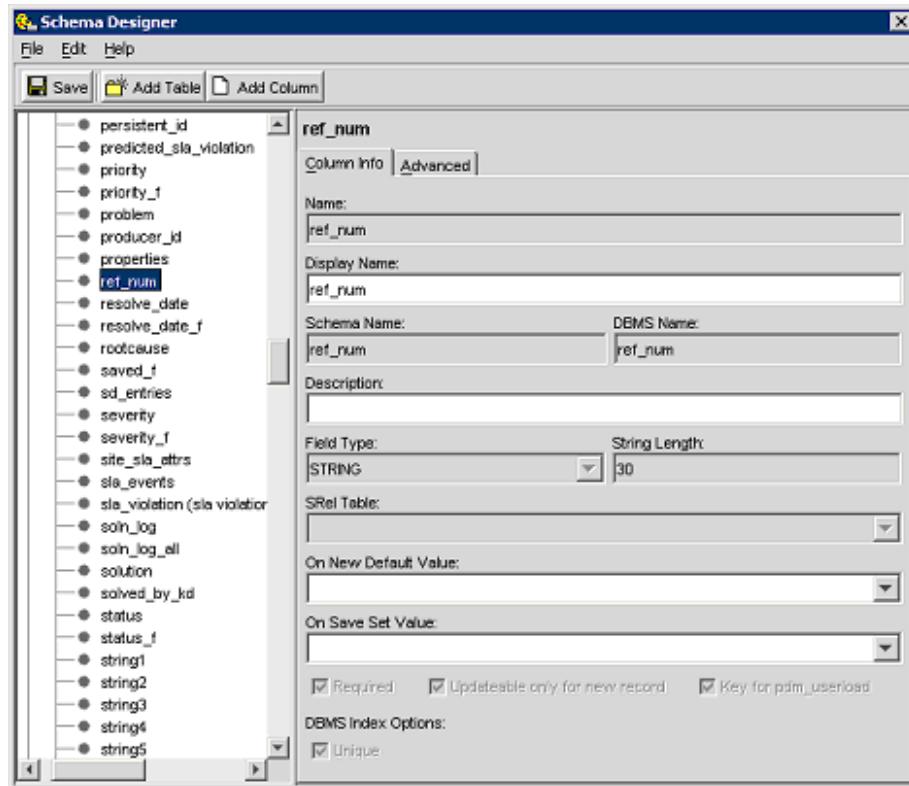
The column stored in the database for a field that references this table. For example, the assignee field of a request is a reference to the Contact table. Since the rel attr of the Contact table is id, the assignee column in a Request contains the id of the referenced contact. You cannot change the value of rel attr.

Function Group

The name of the group that controls the level of access that users have to records in this table. Each contact's access type specifies whether or not they have read, modify, or no access to data in tables in each function group. You can change the value of rel attr by selecting a new value from the drop-down list.

Column Info Tab

In the Tables tree, when you click on a column within a table, WSP populates the Column Info tab with information pertaining to the selected column.



The Column Info tab displays the following information:

Name

The object name of the column. For example, the object name of the Contact alt_phone column is "alt_phone". This is a read-only field.

Display Name

The user-friendly name of the column. You can change the Display Name of a column by entering a new Display Name in this field. For example, the display name of the Contact alt_phone column is "alternative phone".

Schema Name

The name used to refer to the column in Unicenter Service Desk utilities, such as pdm_userload. This field is read-only for standard tables. For site-defined tables, Schema Name defaults to the Object Name. You can change the Schema Name by entering a new value in this field.

DBMS Name

The name used to refer to the table in the physical DBMS. This field is read-only for all tables. For site-defined tables, the DBMS Name is always the same as Schema Name.

Description

A brief description of the column.

Field Type

The data type of the column. This field is read-only for all standard columns in standard tables, and for site-defined columns that have been saved. You can specify or change the field type of new site-defined columns by selecting a value from the drop-down. Field types available are:

INTEGER

Indicates a numeric value.

STRING

Indicates a text string. The number of characters in a string is shown or entered in the String Length field.

DATE

Indicates a date and time. The value stored in the database is an integer containing the number of seconds since midnight on January 1, 1970.

DURATION

Indicates a period of time. The value stored in the database is an integer containing a number of seconds.

DOUBLE

Indicates a real (floating point) number.

SREL

Indicates a foreign key reference to another table. The table referenced is specified in the SRel Table field. The value stored in the database is the rel attr of the referenced table, which can be either an integer or a string. The value displayed on the UI is the common name of the referenced table row.

BREL

Indicates a virtual column representing the set of all objects with an SREL to this table. It exists only in the Object Engine and is not physically stored in the database. This field type should be selected only on direction from a CA employee.

QREL

Indicates a virtual column representing a set of objects selected by the where clause on the Advanced tab. It exists only in the Object Engine and is not physically stored in the database. This field type should be selected only on direction from a CA employee.

LREL

Indicates a virtual column representing a many-to-many relationship between two tables. Details of the relationship are specified on the Advanced tab. You cannot define new LRELS with the Schema Designer; the LREL type is shown only for existing LRELS.

DERIVED

Indicates a virtual column constructed by the Object Engine from the values of other columns, under the direction of a formula specified on the Advanced tab. It exists only in the Object Engine and is not physically stored in the database. This field type should be selected only on direction from a CA employee.

String Length

The length of a string column. This field is blank for non-string columns. It is read-only for all standard columns, and for site-defined columns that have been saved. You can specify or change the length of new site-defined STRING columns by entering an integer between 1 and 32767 in this field.

SREL Table

The table referenced by an SREL column. This field is blank for non-SREL columns. It is read-only for all standard columns, and for site-defined columns that have been saved. You can specify the table referenced by a new site-defined SREL by selecting it from the drop-down list.

On New Default

The default value assigned to this column when a new row of the table is defined. It should be a value appropriate to the field type. Some keyword values are available for particular field types:

NOW

Specifies the current date and time for a DATE column.

USER

Specifies the active user for an SREL to the Contact table.

On Save Set

The value assigned to this column when a row of the table is updated. It should be a value appropriate to the field type. Some keyword values are available for particular field types:

NOW

Specifies the current date and time for a DATE column.

USER

Specifies the active user for an SREL to the Contact table.

Required

When checked, this option indicates that a value must be supplied for the column before a row of the table containing it can be saved. You can set this option for both standard and site-defined columns, and you can disable an option that you have set. However, you cannot turn off the option of a standard column unless it was set by your site.

Updatable only for new record

When checked, this option indicates that a value for this column can be provided only when a row of its table is initially created, and cannot thereafter be changed. You can set this option for both standard and site-defined columns, and you can disable an option that you have set. However, you cannot turn off the option of a standard column unless it was set by your site.

Key for pdm_userload

When checked, this option indicates that this column is one of the columns tested by pdm_userload to determine whether or not its input is an update to an existing row. This option is available only for STRING columns. It is read only for all columns in standard tables.

DBMS Index Options

These options specify characteristics of a column that is an index of the physical DBMS. They are available only for columns in site-defined tables.

Unique

Specifies that the column is unique within the table and that no two rows have the same value for the column.

Ascending

Specifies that the DBMS index is listed in ascending sequence by this column. Mutually exclusive with Descending.

Descending

Specifies that the DBMS index is listed in descending sequence by this column. Mutually exclusive with Ascending

Advanced Tabs

The Schema Designer includes an Advanced tab for both tables and columns. Information on this tab is intended for CA support and field representatives. You will not need to work with this tab for most uses of the Schema Designer, and it will not be discussed further in this document.

Schema Designer Tasks

Modify Tables or Columns

To modify information about a table or column, select the table or column on the Schema Designer by clicking on it, and enter the new information in the appropriate fields. The information you can modify depends on the status of the table or column:

Standard Tables

You can modify the Display Name, Description, and Function Group fields.

Standard Columns

You can modify Display Name, Description fields, the On New Default Value, and the On Save Set value. In addition, if the check boxes for Required or Updatable only for new record are not selected, you can select them. You cannot unselect these options if they are set by default. However, you can reverse your own changes.

Site-Defined Table

If the table is not published, you can modify all fields, except Name, which cannot be changed after the new table has been saved. Once a site-defined table has been published, you can modify only the Display Name, Description, and Function Group fields.

Site-Defined Column

If the column is published, you can modify all fields, except Name, which cannot be changed after the new column has been saved. Once a site-defined column has been published, you can modify only the Display Name and Description fields, the On New Default Value, the On Save Set value, and the check box for Required, Updatable only for new record, Key for pdm_userload, and the DBMS index options.

Add a New Table

To add a new table to the database:

1. Choose Add Table from the Edit menu, or click the Add Table button
The Add New Table dialog appears.
2. Enter the table name in the New Table Name field and click OK. The name of a site-defined table must begin with the letter "z" to prevent conflict with possible future standard tables.
WSP verifies this, and adds a "z" to the beginning of the table name if necessary.
3. Complete the fields in the Table Info tab as appropriate.

Add a New Column

To add a new column to a table:

1. Select the table for which you want to add a column (or select any of its existing columns) and choose Add Column from the Edit menu or click the Add Column button.
The Add New Column dialog appears.
2. Enter the column name in the New Column Name field and click OK. The names of a site-defined column added to a standard table must begin with the letter "z" to prevent conflict with possible future standard columns.
WSP verifies this, and adds a "z" to the beginning of the column name if necessary.
3. Complete the fields in the Column Info tab as appropriate.

Save Changes

To save your changes to the database while you are still modifying tables or columns, choose Save from the File menu or click Save. WSP stores your new or updated schema modifications in the database in either the wsptbl table (for table modifications) or the wspcol table (for column modifications).

Test Schema Modifications

You can test your schema modifications and create, update, and view web forms using them before making any changes to the physical database.

Putting schema changes in Test mode defines them to the Object Engine, but does not physically store their data in the database. Because putting schema modifications in Test mode has the potential of impacting other users, this option is available only if your installation has installed both the `wsp_domsvr` and `wsp_webengine` options to dedicate an Object Engine and Web Engine to WSP.

To put schema changes in Test mode, select Save and set to Test Mode from the File menu on the Schema Designer. This selection saves your changes in the database, and creates a file on the server defining your changes to the Object Engine. This file is called `wsptest.mods`, and is stored in the `site/mods/majic` subdirectory of your Unicenter Service Desk installation directory.

After creating the `wsptest.mods` file, WSP causes its Object Engine to recycle so that it will use the new changes. This may take from a few seconds to a couple of minutes, depending on the complexity of your schema. WSP displays a dialog box while waiting for the Object Engine to recycle, and updates it once the recycle has completed and it is synchronizing its internal storage with the updated Object Engine. When this is complete, WSP displays a message indicating that the Schema has been placed into test mode. When you click OK on this message box, you can begin to use the new schema, including creating and modifying web forms that use it.

The `wsptest.mods` file affects only the Object Engine designated by the `wsp_domsvr` option. Other Object Engines on the same server do not process this file, and the file is not distributed to other servers. In addition, new tables and columns in Test mode are defined to the Object Engine as local objects. This means that the Object Engine knows about them and you can use them on web forms. However, they do not exist in the database, and do not affect other users. Normal Service Desk users do not use the WSP Object Engine, and so are unaffected by the schema modifications you are testing.

Revert Schema Modifications

If you change your mind about your schema modifications after putting them in test mode, you can revert back to the published, version of the schema. Because reverting schema modifications has the potential of impacting other users, this option is available only if your installation has installed both the `wsp_domsvr` and `wsp_webengine` options to dedicate an Object Engine and a Web Engine to WSP.

To revert schema changes from Test mode, select Revert Test Mode from the File menu. WSP deletes the `wsptest.mods`, causing the WSP Object Engine to revert its schema back to the published version.

After deleting the `wsptest.mods` file, WSP causes its Object Engine to recycle so that it can rebuild its internal schema . This may take from a few seconds to a couple of minutes, depending on the complexity of your schema.

Once the Object Engine has completed recycling, the active schema is back to its published version. Note that web forms modified to work with the new schema are not themselves automatically reverted, and may not work correctly when used with the published schema.

Publish Schema Modifications

Once you are satisfied with your schema modifications, you can make them available to all users by publishing them. Publishing modified schema is a two step process:

1. Create or update files describing the modified schema to the Object Engine and to Service Desk utility programs. WSP creates the following files on the web engine designated by the `wsp_webengine` option (which defaults to `web:local`):

wsp.mods

Describes all WSP-maintained schema changes to the Object Engine.

wsp_schema.sch

Describes all WSP-maintained tables and columns.

wsp_index.sch

Describes DBMS indexes for WSP-maintained tables.

wsp.altercol

Names new columns created by WSP but not yet defined to the DBMS.

wsp.altertbl

Names new tables created by WSP but not yet defined to the DBMS. In addition, WSP distributes the `wsp.mods` file to all Unicenter Service Desk servers with an Object Engine.

2. Modify the physical DBMS to contain information about the new schema. This step requires bringing down Unicenter Service Desk services and running the `pdm_publish` script on the primary server.

Important! Step 2 has a significant impact on other users, so you should carefully plan publishing schema changes. We recommend you use Unicenter Service Desk Change Orders to schedule and obtain approval for your planned schema publication.

To begin schema publication, select Save, and Publish from the File menu. This creates the necessary files on Unicenter Service Desk servers, but does not recycle any of them. Thus, the new files have no immediate impact. However, once the files are created, they will be used the next time Unicenter Service Desk services are recycled. Therefore, you should shut down services and run the `pdm_publish` script on the primary server at your earliest convenience after publishing schema modifications.

Once you have completed schema publication with WSP, you cannot make any further changes with the Schema Designer until you have run the `pdm_publish` script. To run `pdm_publish`, shut down Unicenter Service Desk services and enter the `pdm_publish` command at a command prompt.

The pdm_publish command does the following:

- Verifies that there are WSP-produced schema modifications to publish by checking for the existence of the required files in the site mods directory.
- Verifies that you have shut down Unicenter Service Desk services.
- Merges all schema files - WSP maintained and non-WSP maintained - into a single master schema file called ddict.sch.
- Sends the appropriate SQL commands to the DBMS to define the new tables and columns.
- Writes a line to a log file, wsp_schema.log, after each successful DBMS definition of a table or column. In addition to documenting your schema modifications, the log file also serves as a directory to the pdm_publish command itself to allow it to determine which WSP-created tables and columns have already been defined to the DBMS. Therefore, you must not move or modify this file.
- Builds the Unicenter Service Desk data dictionary.

These steps normally take only about a minute. Once they are complete, you can restart Unicenter Service Desk services and begin using your modified schema. If you have created or modified web forms to use the new schema, you should start WSP and publish your new web forms.

Test to Production Migration

One of the design goals for WSP was to make it safe to develop and test schema modifications on a production database. Such features as test mode and dedicated WSP server processes support this goal. However, many users prefer to develop their schema modifications in an independent test system and then migrate them to a separate production system once they are complete. In order to do this:

1. Copy the contents of the wsptbl and wspcol tables from the test database to the production database. We recommend you use the Unicenter Service Desk pdm_extract and pdm_userload utilities to do this.
2. Use WSP on the production system to publish the schema as described in Publishing Schema Modifications above. Using WSP for publishing ensures that all required updates are distributed to all production servers.

Change or Delete Site-Defined Columns after Publishing

Once site-defined schema modifications are published, WSP treats them similarly to standard schema and restricts further changes. Sometimes, it is desirable to delete a site-defined column, or change the length of a site-defined string column. You can accomplish these tasks by manually updating the DBMS and schema outside of WSP, and then running the pdm_wspupd script to update the database wspcol table to synchronize WSP with the external changes. The following procedure can be used to do this:

1. Find the site/mods (UNIX) or site\mods (Windows) subdirectory in your Unicenter Service Desk installation directory.
2. Using any standard text editor, edit file wsp_schema.sch to delete unwanted site-defined columns or change the length of site-defined STRING columns. These are the only changes supported by this procedure.
- Important!** If any of the index options (such as, UNIQUE) were specified for a column to be deleted, use any standard text editor to edit file wsp_index.sch to remove references to the column. If the column was the only indexed column in the table, remove all references to the table from wsp_index.sch.
3. Using any standard text editor, edit file majic/wsp.mods (UNIX) or majic\wsp.mods (Windows) with the same changes made to wsp_schema.sch – that is, delete unwanted site-defined columns, or change the length of site-defined STRING columns.
4. Bring up a command window and issue the command:
pdm_wspupd

The pdm_wspupd script reads wsp_schema.sch and compares it with the wspcol table in the database, writing a line to the console for any differences. The output is similar to:

```
PDM_WSPUPD - Update wspcol table from wsp_schema.sch
Reading wsp_schema.sch to for current DBMS information...
Reading wspcol table for WSP schema information...
String column zSalesOrg.description length changed from 350 to 400
Column zSalesOrg.sym not found in wsp_schema.sch - deleting wspcol row
pdm_wspupd found 1 WSP-maintained column(s) to update and 1 to delete. Please
verify that your DBMS has been manually updated to correspond to
wsp_schema.sch, then reply Y to update wspcol or anything else to cancel.
```

Verify that the changes found by pdm_wspupd correspond exactly to the changes you made to wsp_schema.sch. If they do, type "Y" to confirm the changes. After you confirm the update, the script uses standard Unicenter Service Desk utilities to update the wspcol table. This causes the WSP Schema Designer to show your changes.

5. Stop Unicenter Service Desk services.

6. Using the appropriate utility for your DBMS (such as Ingres Visual DBA), alter the DBMS definition of the columns you changed. You should delete from the database any columns you deleted from `wsp_schema.sch`, and change the database length of any string columns you changed in `wsp_schema.sch`. Take care to ensure that the changes you make to the DBMS correspond exactly to the changes you made to `wsp_schema.sch`.
7. Run `pdm_publish` as described in Publish Schema Modifications above.
8. Start Unicenter Service Desk services.

Chapter 6: Customizing the Web Interface

The Unicenter Service Desk web interface (also referred to as the browser interface) provides you with Unicenter Service Desk functionality through the Internet, including the ability to open, update, or close tickets, display and post announcements, and access supporting data tables. It enables independent browsing of the knowledge base, thereby reducing the number of calls to the service desk and speeding resolution times. The web interface can be fully customized and can be used with many popular web browsers.

If you have installed and configured the web interface according to the instructions in the *Implementation Guide*, you can use the information in this chapter to integrate it into your existing web interface and otherwise customize it to suit your needs. This chapter assumes you are familiar with HTML and the web browser in use at your site.

Note: Web Screen Painter Design view works for Service Desk controls (PDM_MACROS). When working on forms that do not contain SD controls, you can only work on the Source tab. The Employee and Customer web forms do not contain SD controls and therefore appear on the Source tab rather than the Design tab. There are also some Analyst forms that do not contain SD controls, and therefore would appear on the Source tab as well.

Important! Technical support cannot provide help with design or debugging of customizations (this also includes documentation, such as online help systems). For this kind of assistance, please contact Field Services. The information in this chapter is a guide to customizing the Unicenter Service Desk web interface. When doing so, please be aware that you are solely responsible for your own customizations. Unicenter Service Desk technical support can assist you in interpreting and understanding the information in this chapter.

Support for the customization techniques here extends to insuring that the techniques and facilities perform as documented. You should be careful not to exploit undocumented features or to extend documented features beyond their documented capabilities. Such exploitation is not supported and may result in system problems or instability that may appear unrelated to the customization. For this reason, support may ask you to remove customizations to reproduce the problems. Sites should prepare for this eventuality by carefully following the guidelines on placing all modifications in the site mods directory tree and maintaining change logs. Sites that make frequent, complex, or extensive changes should consider approaching Unicenter Service Desk customization as a software engineering project with disciplined source control, testing, and controlled releases to production.

Also, please note that migrating customizations between releases is seldom easy. Unicenter Service Desk development tries very hard to advance the product in ways that preserve the efforts put into customization, but development always assumes the product has been customized only as documented in this guide, particularly with regard to placement of all customizations in the site mods tree. In addition, in the event that Level Two support supplies a patch to a system, the patch is written with these same assumptions. Patching or migrating a system with undisciplined customizations is a risky undertaking that often results in costly system down time. Avoid it by following this guide and practicing sound software engineering principles.

Note: For information on how to secure and configure the web interface, see the chapter "Configure the Web Interface" in the *Administrator Guide*.

Web Screen Painter (WSP)

The primary customization tool for Unicenter Service Desk is Web Screen Painter (WSP). You can install this tool on any Unicenter Service Desk server or client. It provides a simple and easy-to-use user interface that allows you to customize web forms and schema to the requirements of your site without programming. You can use WSP for many tasks including the following:

- Changing field labels.
- Moving fields in a form or changing the appearance of a list.
- Adding fields to a form, or columns to a list.
- Adding a notebook to a form, or changing the tabs in an existing notebook.
- Creating new forms and forms groups.
- Customizing Cascading Style Sheet (CSS) files.
- Previewing your changes in a browser window with your own data before publishing them to other users.
- Adding new tables or columns to the database, or changing the characteristics of existing columns.
- Previewing forms that use customized schema before making any changes to the database.

You can accomplish all these tasks with simple drag-and-drop or point over desired control from the Control Palette and double-click, without any programming, and without even looking at form source code. However, if you want to review and modify source code, WSP also provides a source code editor with keyword highlighting, and seamlessly integrates your source level changes with your design view changes.

However, some of the KT forms cannot be customized in the design view of WSP. For these forms, there are alternate approaches to providing customization such as.

- Document View – the contents of this page are determined by the document template used when creating the document. These templates can be modified on the Administration tab under Documents, Document Templates.
- Knowledge Categories document list – this page can be modified by WSP but is also managed by user preferences. The “Preferences” screen provides personalization per user for defining which document properties should display in the document list and how many documents should display per page.

Modifying schema by adding new tables and columns requires administrator authorization, and is discussed in the chapter Customizing the Schema Using the Web Screen Painter (*see page 81*). The remainder of this chapter discusses form customization.

Start Web Screen Painter

In Windows, you can start Web Screen Painter from the Start menu in two different ways:

Choose Programs, CA, Unicenter Service Desk, Web Screen Painter

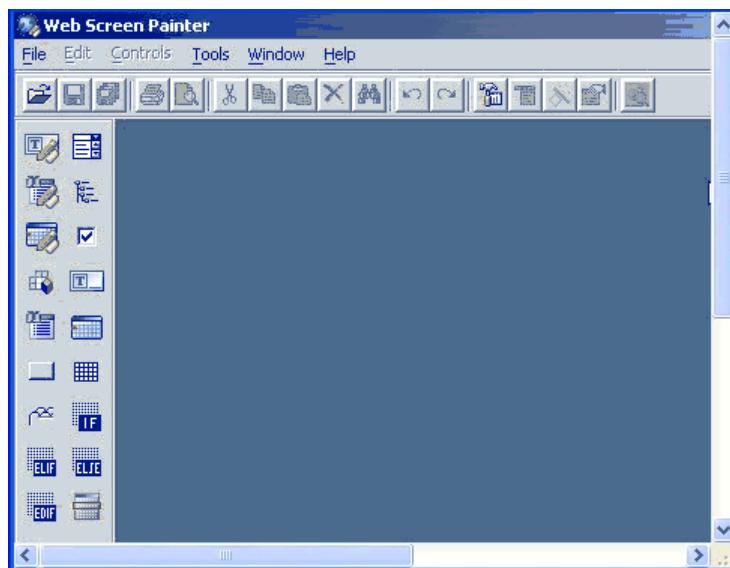
OR

Choose Run and enter pdm_wsp.

In Linux, you can start Web Screen Painter by entering the command `pdm_wsp` with `$NX_ROOT/bin` in your path.

Note: Linux users must have Mozilla installed in order to use WSP.

WSP displays a standard Unicenter Service Desk login form in a browser window. After you login, WSP displays its main form:

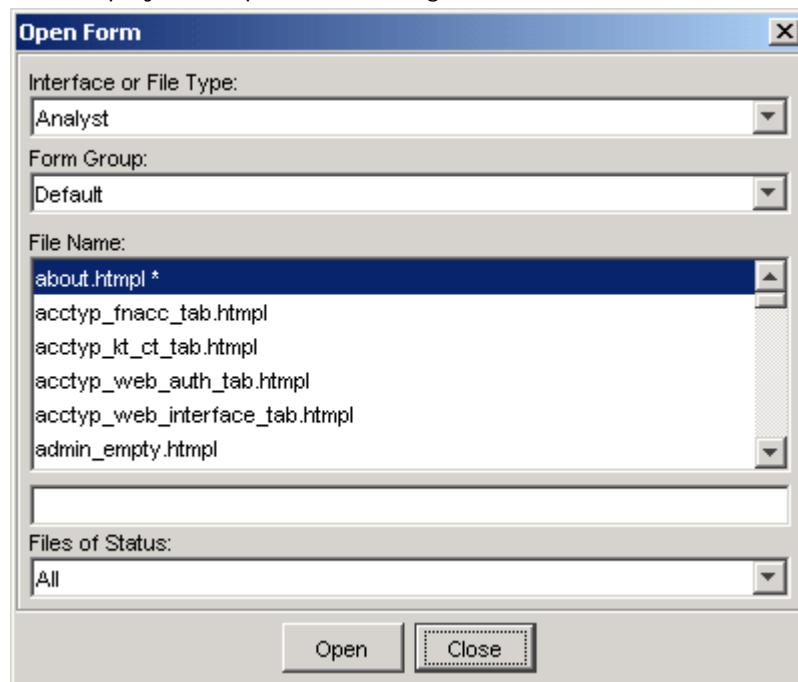


Open a Form for Editing

To begin editing a form:

1. Choose Open from File menu or click  on the toolbar.

WSP displays the Open Form dialog:



2. Select the Interface (Analyst, Customer, Employee, Default) or File Type (CSS Stylesheet, JavaScript, or HTML) and the forms group that contains the form you want to edit.
3. Select either the form you want from the scrolling list, or enter its name in the textbox.

WSP automatically scrolls the list to the first name matching the characters typed when you enter a name in the textbox.

You can use the Files of Status drop-down list to restrict the list of files displayed:

Site Modified with Unpublished Changes (+)

Restricts the list to files that have been modified with WSP, but not yet published. These files are identified with a plus sign (+) after the file name.

Site Modified (*)

Restricts the list to forms modified at your site, both published and unpublished. Unpublished files are identified with a plus sign (+) after the file name; published site modifications are identified with an asterisk (*) after the file name.

All

Shows the list with no restrictions. Unpublished files are identified with a plus sign (+) after the file name; published site modifications are identified with an asterisk (*) after the file name.

Create a New Form

To create a new form:

1. Choose New from the File menu.

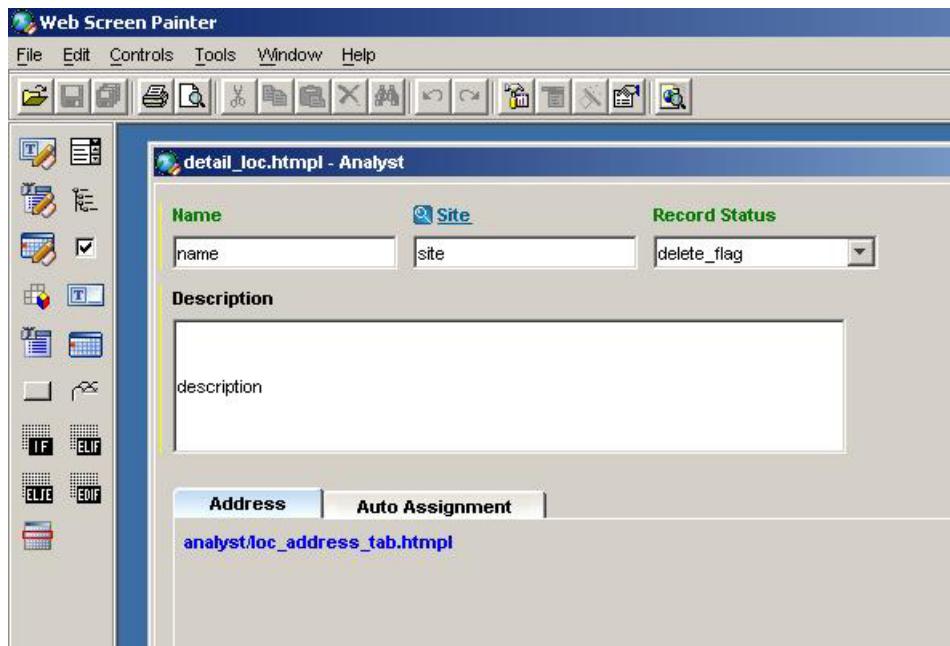
WSP displays the New Form dialog:

2. Select an Interface and Forms Group for the new form, and whether the form is to be a detail form, list form, or menu bar.
3. Select a factory (or table) for the new form.

Note: There can be only one detail or list form per table in a forms group, so you should edit an existing form (rather than create a new one) for tables that already have an existing form. If you want to have multiple versions of a form, create one or more form groups to hold the additional versions as described in Creating Web Form Groups (see page 130).

Form Edit Window

Once you've opened an existing form or asked to create a new one, WSP displays the Form Edit window:



There are two tabs in the edit window, the Design tab and the Source tab. The Design tab is available for detail forms, list forms, and menu bar forms, and shows the controls on the form laid out more or less as a user would see them. It is not an image of how the form looks to an end user; to see this, select Preview from the Tools menu or click  on the toolbar.

The Source tab is a Notepad-style editor allowing you to review and edit the source code for a form. Some forms are editable only in the Source tab. For those forms, the edit window opens up on the Source tab, and the Design tab is disabled.

The title bar of the edit window shows the name of the form, its interface, and (if appropriate) its form group. You can open edit windows for more than one form at the same time.

Edit List and Detail Forms in Design View

The Design view tab shows the controls on a form, arranged in the same tabular form they would be displayed to a client. You can rearrange controls dragging and dropping them. You can delete a control by:

- clicking it and selecting Delete from the Edit Menu.
- or clicking  (delete).
- or  (cut) on the toolbar.

Design view is used for editing controls. It does not show the form as it would be viewed by an end user (to view a form as it would be viewed by the end user, use Preview from the Tools menu or click  on the toolbar). The principal differences between Design view and the end user view are:

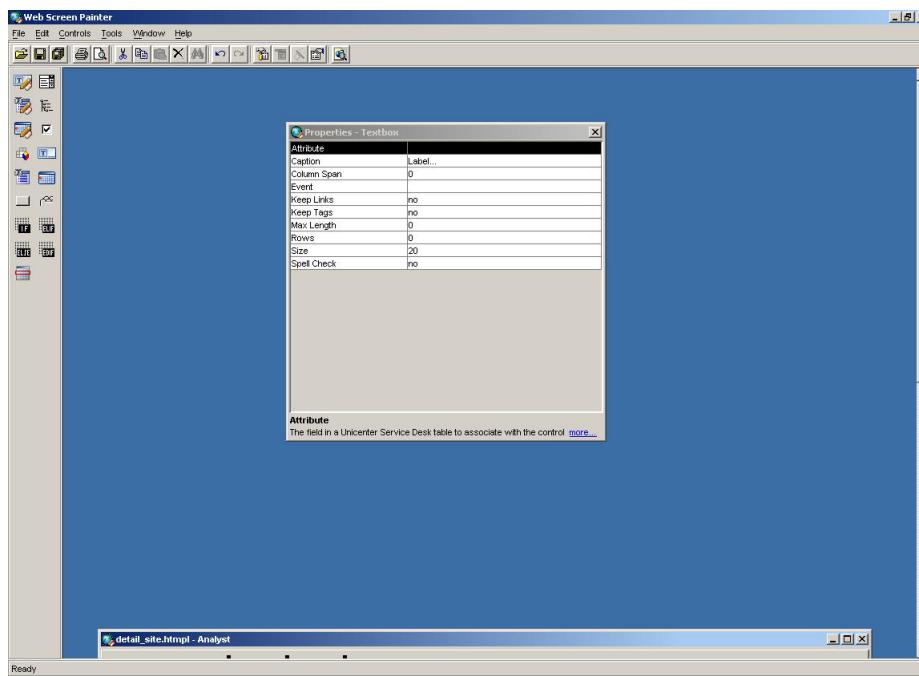
- Fonts and styling are not used in Design view.
- Each control shows its associated attribute name in Design view.
- WSP shows only Unicenter Service Desk controls (those defined by <PDM_MACRO> statements). It does not show content defined by standard HTML tags or JavaScript.
- WSP shows all controls on the form, regardless of conditionals (PDM_IF statements). This allows you to edit anything that might be on the form. WSP shows conditional controls (see page 114) themselves as red text, such as If or Else.

Properties Dialog

To change the properties of a control (including its label), display its Properties dialog. There are several ways to do this:

- Select a control on the form and choose Properties from the Controls menu
- Double-click a control on the open form
- Select the control and then select  on the toolbar
- Press F4
- Right-click on the control or form and select Properties from the drop-down list

Any of these actions displays the Properties dialog for the control. For example, the textbox properties dialog is:



All Properties dialogs contain fields for the Attribute (column) name; for the Caption (label), and for the Column Span (number of columns in the grid). The remaining fields in a Properties dialog vary with the type of control - see the *Web Screen Painter* online help for details. You can display the online help for a Properties dialog by displaying the dialog and clicking F1.

To change a value in a Properties dialog, simply type the new value in the appropriate place. Changes take effect as soon as you click outside the field, as well as when you close the Properties dialog.

WSP displays a brief summary of the significance of a property in a note that appears at the bottom of the Properties form when you select the property. For more detailed help, press the F1 key.

Insert a Control

There are several ways to insert a control on a form:

- Drag-and-drop the desired control from the Control Palette on the left side of the main WSP window to the desired spot on the form.
- Click a position on the form where you want to place the new control and select the desired control from the Control menu.
- Copy an existing control and paste it to the form.

Once the new control is properly placed, display and edit its properties as described in the previous section.

The controls that can be inserted on both list and detail forms are:

Control	Icon	Description
Insert Row		Causes the selected control to be the last in its current row (moves following controls to the next row).
Delete Row	N/A	Deletes all controls on the same row as the currently selected control.
Textbox		Inserts a single or multi-line textbox for editing a string or text field.
Dropdown		Inserts a drop-down selector for editing a field validated against a table.
Lookup		Inserts a lookup control for editing a field validated against a table. The control consists of a textbox with a hyperlink in the label that pops up a select form.
Button		Inserts a button.
Hierarchical Lookup		Similar to a Lookup control, except that it is used for a field with a hierarchical selector (such as request category).
Date		Inserts a date field. The control consists of a textbox with a hyperlink in the label that pops up a date selector.

The following additional controls are available for detail forms only:

Control	Icon	Description
Checkbox		Inserts a check box.
HTML Editor		Inserts an HTML editor for a text field that contains HTML.
Read Only Textbox		Inserts a non-editable text field.
Read Only Lookup		Inserts a non-editable lookup field. The field is displayed as a hyperlink to pop up the detail form defining it.
Read Only		Inserts a non-editable date field.
Notebook		Inserts a notebook. There can only be one notebook on a detail form, so this control can be inserted only on forms that do not already contain a notebook.

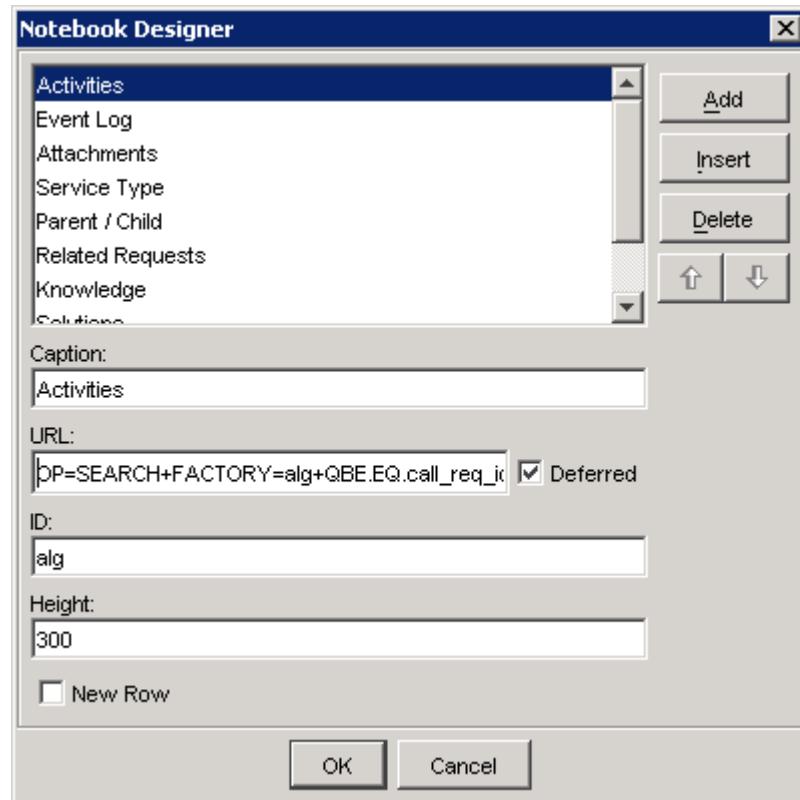
The following additional control is available for list forms only:

Control	Icon	Description
List		Inserts a list. There can only be one list on a list form, so this control can only be inserted on new list forms.

Notebook Designer

Many detail forms contain a notebook with two or more tabs. You can also use the Notebook control to add a notebook to a detail form that doesn't already contain one.

The Properties dialog for a Notebook control is replaced by the Notebook Designer. To open the Notebook Designer, double-click the Notebook control. WSP displays the Notebook Designer:



The Notebook Designer allows you to add, insert, and delete notebook tabs and to change their captions. You can also use the arrow buttons (\uparrow and \downarrow) to rearrange tabs by changing the position of the currently selected tab. The New Row check box specifies whether or not the selected tab begins a new row in the notebook header. See the *Web Screen Painter* online help for explanations of the other fields on this form.

Unicenter Service Desk supports two types of notebook tabs:

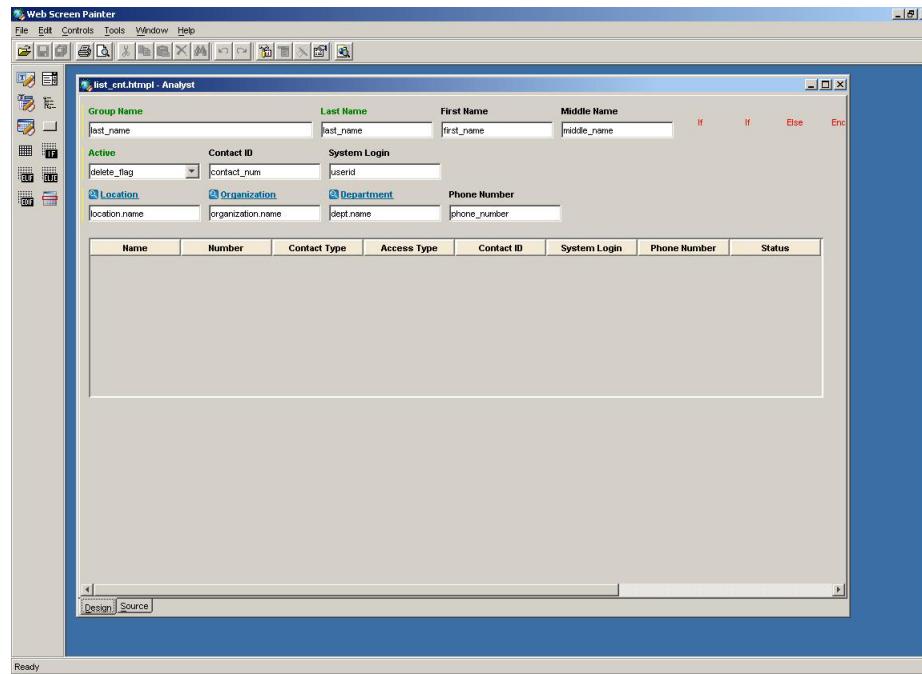
- A *deferred tab* is loaded only when selected by the user in the notebook header. Its Notebook Designer entry specifies a URL. This URL can be either a standard Web URL beginning `http://` or a Unicenter Service Desk URL beginning `OP=`. To specify a deferred tab, check the Deferred check box.

- A *standard tab* is loaded at the same time the form is loaded. Its Notebook Designer entry specifies the file name of an HTMPL file that defines the contents of the tab, which must be bracketed by <pdm_form> tags in the file. To specify a standard tab, clear the Deferred check box.

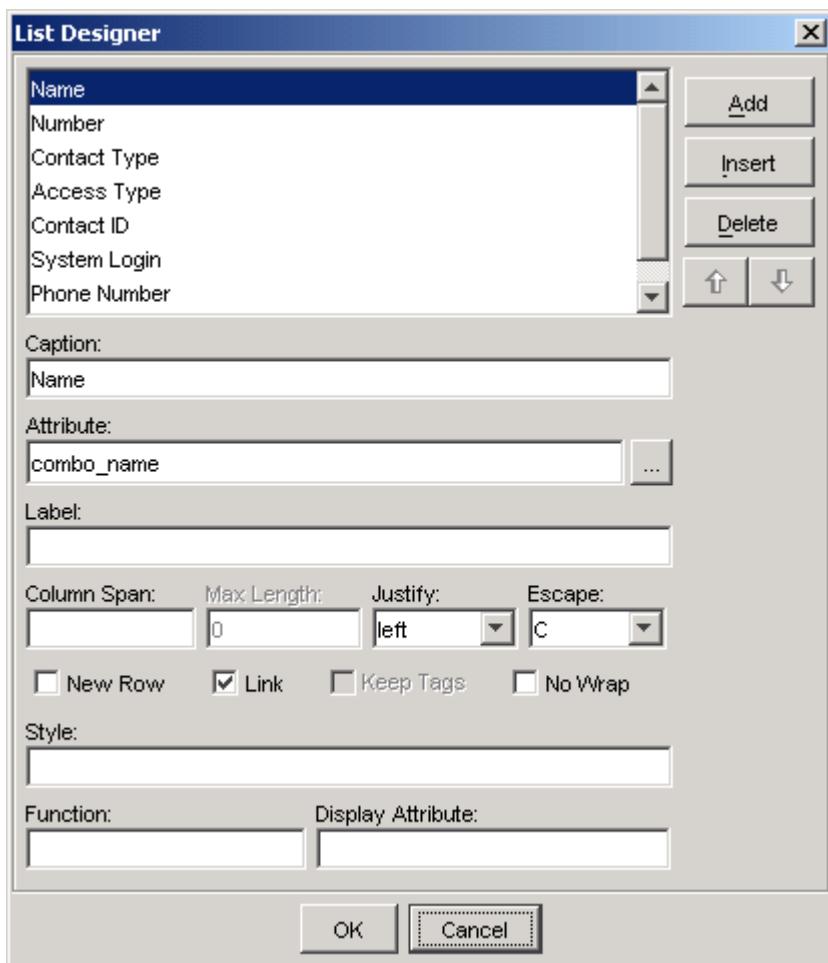
The contents of a standard tab are defined in a separate HTMPL file. To edit this file with WSP, double-click the hyperlink file name that WSP shows on the body of the tab in Design view. WSP opens another form edit window for the HTMPL file defining the tab.

List Designer

Unicenter Service Desk list forms normally consist of a search filter at the top of the form, and a list at the bottom of the form. The search filter section of a list form resembles a detail form, and you edit it in the same way. The only difference is that fewer controls are available in this window. The list forms support only the textbox, dropdown, date, lookup, command button and hierarchical lookup controls.



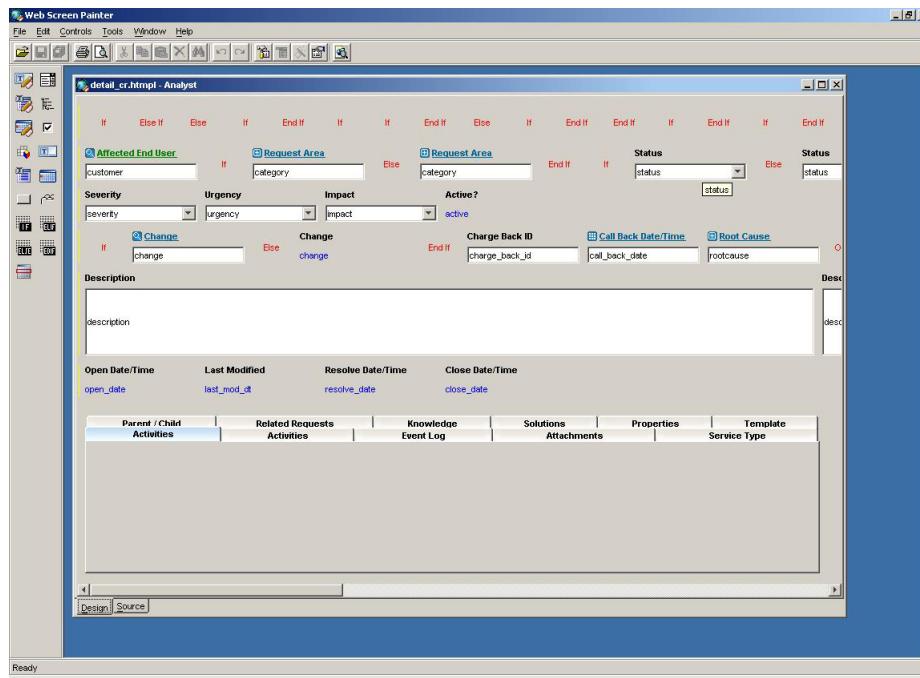
WSP displays the list section of a list as an empty rectangular box with the list headings at the top. The Properties dialog for a List control is replaced by the List Designer. To display the List Designer for a list, double-click the list control. WSP display the List Designer:



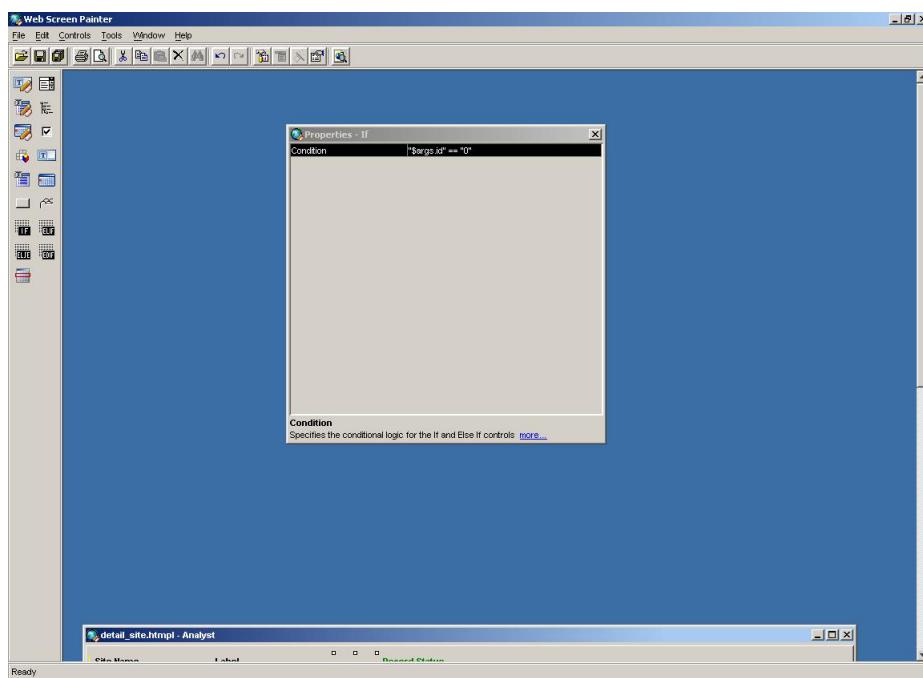
The List Designer allows you to add, insert, and delete attributes and to change their captions (column headers). You can also use the arrow buttons (and) to rearrange attributes by changing the position of the currently selected attribute. The New Row check box specifies whether the selected attribute begins a new row of attributes in the data for a single database row. See the *Web Screen Painter* online help for explanations of the other fields on this form.

Conditional Controls

Unicenter Service Desk supports conditionally including content on a form. For example, the Request Detail form (detail_cr.htmpl) uses a lookup control for its change attribute only for new requests. For existing requests, it uses a read-only lookup. WSP shows this in Design view as follows:



WSP shows both Textbox controls for the change attribute side by side, even though the end user will only see one at a time. It shows the conditional control as a word in red text, such as If or Else in the example. You can view the Properties dialog for a conditional control the same way as you display the Properties dialog for a normal control. For example, double-clicking on the word If in the example displays:



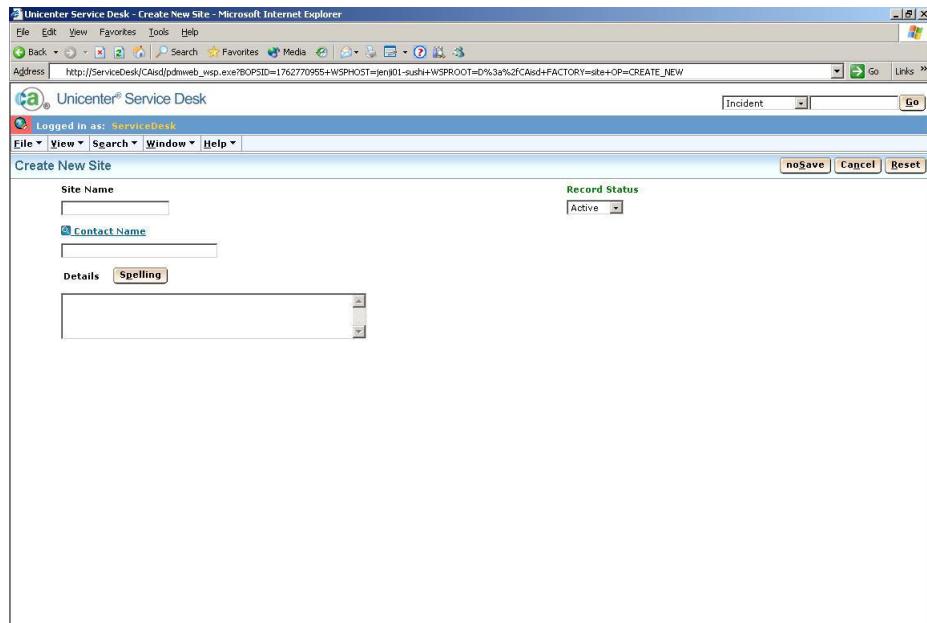
The conditional controls shown by WSP correspond to PDM_IF and its associated tags. PDM_IF: Conditional Processing discusses these tags, including the syntax of a conditional.

WSP shows four types of conditional controls:

Control	Icon	Description
If		Begins a conditional.
Elif		Optional; specifies an alternative condition (else if). There can be any number of elif controls.
Else		Optional; specifies an alternative. If provided, must be the last control before endif.
Endif		Required; ends the conditional.

Preview Forms

To see how a form would look to an end user, choose Preview from the Tools menu, or click  on the toolbar. WSP copies the modified form to the server, where it is stored in a directory accessible only for your WSP preview requests, and then submits a URL to pop up the form in a browser window.



A WSP preview window allows you to view a form as it would be seen by an end user. Although it resembles a standard Unicenter Service Desk window, and most buttons and menus are functional, it is not a standard session, and you should not attempt to use it that way. The following are the principal limitations of a preview session:

- It is normally read-only. That is, although you can try out functions such as editing data, all database update requests are ignored; you cannot change the database in a WSP preview session. The browser window indicates this in two ways:
 - The red WSP icon on the top left corner indicates that the browser window shows a read-only preview session. It is possible for the Unicenter Service Desk administrator to allow updates in the preview window. However, this is not recommended. If your administrator has configured WSP this way, the icon is yellow (caution).
 - Any occurrence of the word "Save" on a button or menu is changed to "noSave" to indicate that no database update will occur.

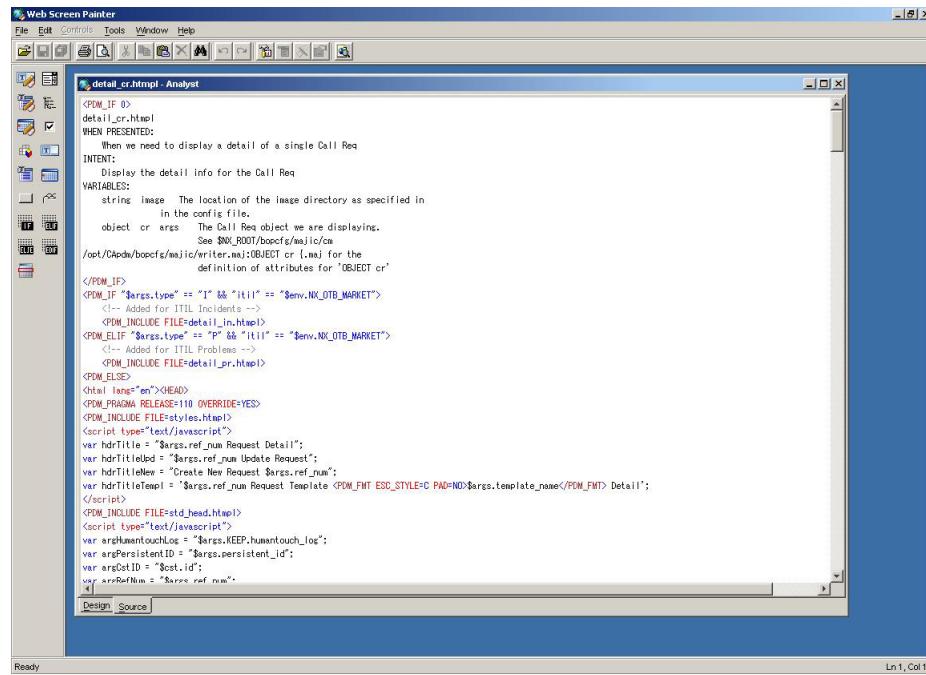
- Not all functions are available. WSP preview always shows you the form or tab you are working on. However, many forms are intended to be reached by a specific path through the application, and their environment may not be set up correctly when they are shown directly. If you click a button or attempt to use a feature that has not been properly set up, WSP displays a message explaining that the function is not available in preview mode.
- WSP always shows a detail form in edit view, and populates it with data from your database (it uses the most recently added row from the appropriate table that you are authorized to view). To see the read-only view of the form, click the noSave button.
- WSP always shows a list form listing a single row from the database, with its search filter closed. You can view and change the search filter and repeat the search as necessary to preview the form.

The default behavior for WSP preview is to display a detail form in edit view, or a list form in list view. You can modify this behavior for a particular HTMPL form with the PDM_WSP tag as described in the PDM_WSP: Control WSP Preview.

Edit in Source View

Sometimes it is necessary or useful to look at source code for a form. This can be helpful for editing forms other than list or detail forms, or for editing HTML or JavaScript form elements that are not shown in Design View.

To switch to Source view for a form, click the Source tab. WSP displays the source code for the form:

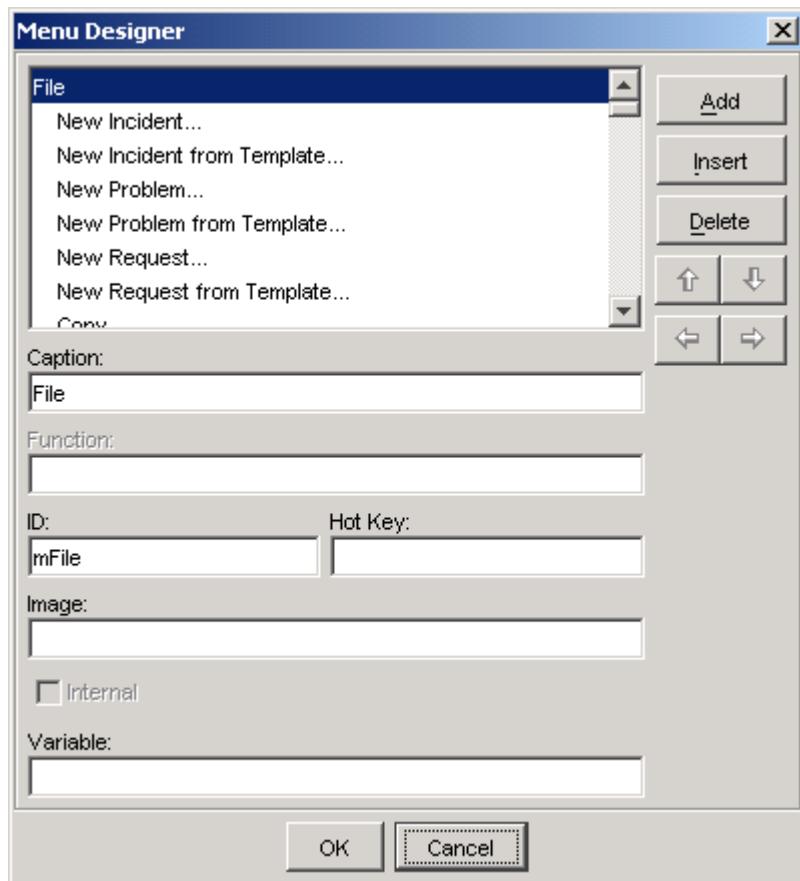


If a control is currently selected, WSP automatically moves the cursor to the beginning of the source code that defines that control.

The Source view editor is a basic text editor, similar to the standard Windows Notepad editor, except that Source view is color-coded. HTML and other keywords are highlighted and colored. You can control the font and the color coding used in Source view by choosing Options from the Tools menu. The Options dialog shows the font used in Source view and the default color for eight HTML and JavaScript elements. To change a color, click on the ellipsis button next to it and select the desired color from the palette.

Edit Menu Bars

Forms with names beginning "menubar_" define a menu bar. The Design view for a menu bar shows the menu at the top. You can click on a menu item to lower the menu, but cannot otherwise edit the menu bar directly in Design view. To edit a menu bar, double-click the menu item. WSP displays the Menu Designer:



Note: Menus (and menubar forms) are used only in the analyst interface. The customer and employee interfaces use a "launch bar" containing actual links, not dropdown menus. To customize the customer or employee launch bar, edit form std_body_site.htmpl from the appropriate interface.

The Menu Designer allows you to add, insert, and delete menus and their items and to change their captions. You can use the up and down arrow buttons (\uparrow and \downarrow) to rearrange menus and items by changing the position of the currently selected item.

Both the Add and Insert buttons insert a new menu item: Add places the new menu item at the end of the menu, while Insert places it before the currently selected item.

To insert a new item on the menu bar itself,

1. Add or insert a menu item.
2. Click the left arrow button () to convert it to a menu bar item.
3. Click the right arrow button () to if you want to reverse this action.

See the *Web Screen Painter* online help for explanations of the other fields on this form.

Edit Stylesheets

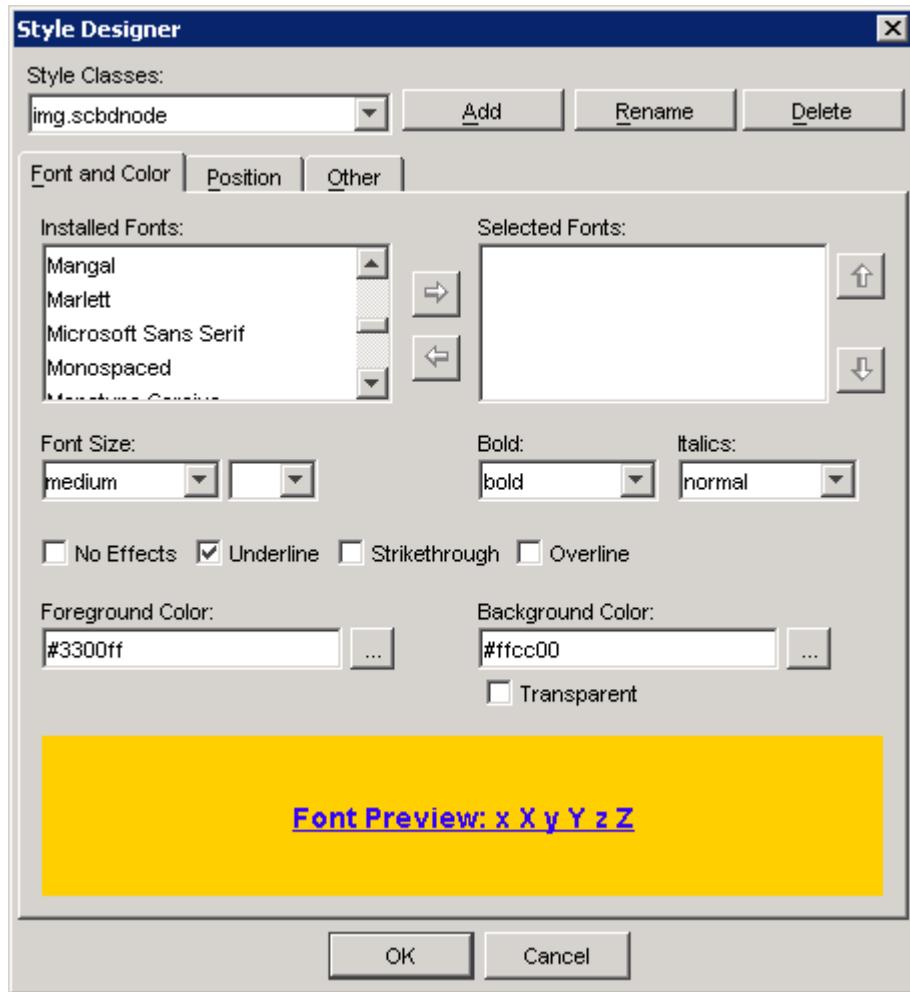
You can use WSP to edit CSS (cascading stylesheet) files. To edit a stylesheet:

1. Choose Open from the File menu or click  on the toolbar.
WSP displays the File Open dialog.
2. Select CSS Stylesheet from the Interface or File Type drop-down.
A list of stylesheets is displayed.

Or to create a new stylesheet.

1. Choose New from the File menu.
The New Form dialog appears.
2. Select CSS Stylesheet from the Interface or File Type drop-down list and click New.

In either case, WSP displays the Source view of the stylesheet. You can edit directly in Source view, or bring up the Style Designer by choosing Style Designer from the Tools menu or clicking  on the toolbar.



The top section of the Style Designer allows you to control the classes within the stylesheet. The Style Classes drop-down in the upper left of the Style Designer allows you to select a class to edit. The Add, Rename, and Delete buttons allow you to create a new class, or rename or delete an existing one.

There are three tabs on the Style designer. The Font and Color tab allows you to select attributes for text formatted by the style class, and preview how it will look. The Font Preview section at the bottom of this tab shows you how the style will look.

The Position tab allows you to control positioning, and the Other tab allows you to control visibility, display, overflow, and cursor attributes. Note that there are a number of style attributes, such as margin and border that can neither be seen nor edited in the Style Designer. These must be edited in Source view.

When you click OK in the Style Designer, WSP reformats the stylesheet and updates the Source view. You can continue to edit in Source view or bring up the Style Designer again.

Note: For performance reasons, Unicenter Service Desk stylesheets are delivered in two forms: Individual files (such as search_filter.css) and combination files grouping a number of individual files with comments and excess white space removed (such as analyst_styles.css). WSP always edits the individual files; you cannot edit a combination file directly. When you publish stylesheet changes, WSP automatically builds the associated combination file if necessary.

Edit HTML and JavaScript

You can use WSP Source view to edit HTML and JavaScript forms. To edit either, choose Open from the File menu or click  on the toolbar. WSP displays the File Open dialog. Select HTML or JavaScript on the Interface or File Type drop-down list to display the list of files available to edit.

Note: For performance reasons, some Unicenter Service Desk JavaScript files are delivered in two forms: Individual files (such as window_manager.js) and combination files grouping a number of individual files with comments and excess whitespace removed (such as std_head.js). WSP always edits the individual files; you cannot edit a combination file directly. When you publish script changes, WSP automatically builds the associated combination file if necessary.

Saving Changes

At any time, you can save changes you have been editing. To save changes to a particular file, select its edit window, and then choose Save from the File menu, or click  on the toolbar. To save changes to all files you are editing, choose Save All from the File menu or click  on the toolbar.

Note: WSP always saves changes on the server, not on your local PC (unless your local PC is the server). When you save a file, it becomes accessible to other WSP users in a preview session, but is invisible to normal Unicenter Service Desk users. This is because WSP saves all files in the site/mods/wsp directory (UNIX) or the site\mods\wsp directory (Windows), and this directory is not used by a normal Unicenter Service Desk session.

Deleting Changes Prior to Publication

If you are not satisfied with changes you have made, you can delete them prior to publication. Deleting changes has the effect of deleting a new form or leaving an existing form in its current state. To delete changes, select Delete Form from the File menu.

Requests to delete a form take effect when you publish changes. You can cancel a pending delete request by selecting Undelete Form from the File menu. You cannot cancel changes after publication; the only way to change a published form is to edit it again.

Deleting Forms after Publication

Only site-modified forms can be deleted. Requests to delete a previously-published form take effect when you publish changes. You can cancel a pending delete request by selecting Undelete Form from the File menu. You undo changes to a form after publication; the only way to change a published form is to edit it again.

Publishing Changes

When you are satisfied with changes, you can make them available to all Unicenter Service Desk users by publishing them. Publishing updates all Unicenter Service Desk servers with new or revised forms.

To publish changes, choose Publish from the File menu. If you have any unsaved changes, WSP prompts you to save them, and then pops up a confirmation dialog showing all pending WSP changes (including those saved in previous sessions, or saved by other WSP users). By default, all changes are selected for publication. You can change the selection of changes to be published by clicking on them or on the Select All or Select None buttons. When you are satisfied with the selection, click OK. WSP makes the selected changes available to all users.

Test to Production Migration

One of the design goals for WSP was to make it safe to develop and test forms modifications on a production database. Such features as a WSP-only directory tree on the server, dedicated WSP server processes, and read-only preview sessions support this goal. However, many users prefer to develop their forms modifications in an independent test system and then migrate the forms to a separate production system once they are complete. In order to do so:

1. Copy any HTMPL forms to be migrated from the appropriate subdirectory of site/mods/www/htmpl on the test system to the same subdirectory of site/mods/wsp/project on the primary server of the production system.
2. Copy any CSS, JavaScript, and HTML files to be migrated from the appropriate subdirectory of site/mods/www/wwwroot on the test system to the same subdirectory of site/mods/www/wwwroot/wsp/project on the primary server of the production system.
3. Use WSP on the production system to publish the forms as described in *Publishing Changes* above. Using WSP for publishing ensures that the new or updated forms are distributed to all production servers.

You can use any file copying method supported by your operating system to perform the copying described in steps 1 and 2 above. Windows users should substitute backslash (\) for slash (/) in the directory paths shown.

HTML Templates (HTMPL Form)

Overview

Forms in the Unicenter Service Desk web interface are delivered as HTML templates, in files with a suffix of .htmpl. These are called HTMPL forms in the remainder of this document.

An HTMPL form contains standard HTML (including JavaScript) plus language extensions that are interpreted by a Unicenter Service Desk server daemon (or service) called the web engine that delivers standard HTML to the browser. These extensions are:

- References to server variables (see page 156). These are indicated by a name beginning with a dollar sign. They can be the values of columns in the Unicenter Service Desk database, references to web engine configuration properties, or other server information.
- Special tags directing the web engine to perform tasks on the server, such as read information from the Unicenter Service Desk database. These tags have names of the form <PDM_...> or <pdm_...>, listed in HTMPL Tags (see page 131).

These extensions will be discussed in more detail later in this document.

Note: You do not need to understand the HTMPL extensions or even HTML itself to be able to customize Unicenter Service Desk forms with Web Screen Painter.

Template Naming Conventions

The following naming conventions are used to identify the four basic types of HTMPL files, where *xxx* is the object:

Template Type	Name
List (search filter and results)	list_ <i>xxx</i> .htmpl
Combined read-only and edit detail form (analyst detail interface)	detail_ <i>xxx</i> .htmpl
Read-only detail form	detail_ <i>xxx</i> _ro.htmpl
Edit detail form	detail_ <i>xxx</i> _edit.htmpl

You can find the definitions of the objects and their properties in \$NX_ROOT/bopcfg/majic/*.maj (UNIX) or *installation-directory*\bopcfg\majic*.maj (Windows). The appendix "Objects and Attributes" (see page 461) lists all the objects and attributes that define Unicenter Service Desk.

HTMPL Directories

The Different Web Interfaces section in the "Configure the Web Interface" chapter of the *Administrator Guide*, describes the web interfaces supplied with Unicenter Service Desk. There are different sets of HTMPL files supplied to implement these interfaces, as shown in the following table:

Operating System	Directory Containing HTMPL Files
Windows	<i>Installation-directory</i> \bopcfg\www\htmpl\web\interface
UNIX	\$NX_ROOT/bopcfg/www/htmpl/web/interface

In this table, *interface* is the name of the interface (analyst, customer, or employee).

Note: There is no separate directory for guest interface files; by default, this interface uses the employee interface files. You can change the guest user interface by changing the access type associated with user System_Anonymous. Both the customer and employee files dynamically modify themselves depending on whether the current user is a known user or a guest, using the <PDM_IF> template command described in this document.

There are three additional interface subdirectories under the htmpl directory:

default:

Contains HTMPL files common to all interfaces. When searching for a file, the web engine looks first in the directory corresponding to the current user's interface, and then in the default directory.

pda/analyst: (UNIX)

pda\analyst: (Windows)

Contains HTMPL files used by the mobile device interface. In Unicenter Service Desk r11, the mobile device interface is provided only for analysts.

web/interface/legacy: (UNIX)

web\interface\legacy: (Windows)

Contains HTMPL files from your previous release of Unicenter Service Desk that are no longer used. This directory is automatically created if you migrate from a previous release when you install Unicenter Service Desk. You can delete the legacy directory when none of its files are referenced by your customized files.

We strongly recommend that you do not directly modify the supplied HTMPL files. Instead, either use Web Screen Painter, or manually copy the file you want to modify to the site mods directory, and modify it there. The Unicenter Service Desk web server looks for a new form in the appropriate site mods directory before checking the distribution directory. The standard site mods directories for each of the interfaces are as follows:

Operating System	Directory For Site-Modified HTMPL Files
Windows	<i>installation-directory\site\mods\www\htmpl\interface\interface</i>
UNIX	\$NX_ROOT/site/mods/www/htmpl/interface/interface

Note: If you change the form and save it into the *install directory\site\mods\www\htmpl\interface* directory, the form will be seen by everyone, regardless of the form group to which they belong. If you save it into the *install directory\site\mods\www\htmpl\interface\interface* directory, only those Contacts that are defined as belonging to that form group will see the changed forms.

In the previous table, *interface* is the name of the interface (analyst, customer, or employee). There is no separate directory for guest interface files; this interface uses the employee interface files. The advantage of storing your modified HTMPL files in the site mods directory is that this directory is preserved when you install Unicenter Service Desk maintenance or a new release. In addition, keeping your modified files in site mods while preserving the originals ensures that you always have a correct copy of the originally distributed HTMPL file.

Each web interface page has a primary function, as indicated in the following table that lists the major HTML templates. However, you can add <PDM_FORM> blocks to any template to directly access any web interface supported operation. For example, you can modify the main menu to include fields for submitting an issue without using the intermediate page, or you can add search criteria fields and a search button to a list form:

Web Page	HTML Template
Main form	menu_frames.htmpl
Display/create/update a change order	detail_chg.htmpl
Display a list of change orders	list_chg.htmpl
Display/create/update and issue	detail_iss.htmpl
Display a list of issues	list_iss.htmpl
Display/create/update a request	detail_cr.htmpl
Display a list of requests	list_cr.htmpl
Display announcement detail information	detail_cnote_html
Display a list of announcements	list_cnote.html
Login	login.htmpl

Note: For a complete list of templates, view the contents of the directories in the table at the beginning of this section.

Creating Web Form Groups

You can collect customized web pages into one or more form groups. Form group directories are in the following directories:

In Windows:

```
install-directory\site\mods\www\htmpl\web\interface  
install-directory\site\mods\www\wwwroot\subdirectory
```

In UNIX:

```
$NX_ROOT/site/mods/www/htmpl/web/interface  
$NX_ROOT/site/mods/www/wwwroot/subdirectory
```

Each form group is a subdirectory under these directories. You specify the customized form directory in the Customization Form Group field of the access type.

When a user requests a form, the web engine looks first in the appropriate customized form group directory, then in the standard directory for the user's web interface, and finally in the default directory. You can define more than one access type for the same web interface, each with a different customized form group. This allows you to define a few specialized forms for different types of users, and still take the majority of the forms from the standard interface.

A similar process occurs when a web page requests a file from one of the subdirectories of wwwroot (css, html, img, or scripts). The webengine examines an HTMPL reference of the form CAisd/img/xxx.gif and converts it to one of:

- /CAisd/sitemods/img/formgroup/xxx.gif
- /CAisd/sitemods/img/xxx.gif
- /CAisd/img/xxx.gif

selecting the first one where it finds xxx.gif.

The procedure for setting up a new web form group is as follows:

1. If you want a form group besides the predefined Analyst, Customer, or Employee form groups, create a new form group by selecting Save As from the File menu in Web Screen Painter and clicking the Add Form Group button on the Save Form As dialog. For example, if you want to provide two separate customized versions of the Analyst interface, you might create form groups called Analyst1 and Analyst2 to handle these. You might also define a new form group if the interface you are defining does not logically fit into one of the predefined form groups.

2. On the web client (not a WSP preview session), select Security, Access Types from the Administration menu. Then click on an access type (or create a new one) and use the Customization Form Group drop-down list on the Access Type Detail window to assign a form group to an access type. Unicenter Service Desk determines the access type when a contact logs in and uses customization form group to determine where to look in the site mods directory structure for customized forms. If the web engine does not find a form in the form group directory, it looks first in the standard directory for the user's access type, and then in the default directory.
3. In WSP, select Save from the File menu, or manually copy the customized HTMPL files to the following directory:

In Windows: *installation-directory\site\mods\www\htmpl\web\form_group_name* directory

In Unix: \$NX_ROOT/site/mods/www/htmpl/web/form_group_name

Once you have set up a new web form group and copied any supporting files to the appropriate subdirectories, you must restart the web service before the changes take effect.

HTMPL Tags

PDM_EVAL: Insert the Value of a Pre-Processor Variable

The pdm_eval tag is used to insert the value of a pre-processor variable into the input of the webengine parser. If used inside a macro, its effect is deferred until the macro completes.

The pdm_eval tag works similarly to pdm_include or pdm_macro. It inserts the text into the parser at the point of the tag, exactly as if the value of its variable had been coded in place of the tag.

pdm_eval has the following syntax:

<PDM_EVAL TEXT=PRE.name>

name

(Required) The name of the pre-processor variable whose value is to be inserted into the webengine's input.

PDM_FORM: Start an HTML Form with a Session ID

<PDM_FORM> and </PDM_FORM> can be added to any web interface HTML template to create an HTML form including two hidden fields for the server variables SID (session ID) and FID (form ID). The optional OP operand creates an additional hidden field for one of the supported operations, as with the PDM_LINK tag. Except for the automatically-generated hidden fields, <PDM_FORM> and </PDM_FORM> are used in the same way as the standard HTML <form> and </form> tags (and generate these tags as part of their expansion).

PDM_FMT: Format Text from a Server Variable

The <PDM_FMT> and </PDM_FMT> tags are used to format blocks of text inserted by server variables (\$args.xxx) as directed by its arguments.

Note: <PDM_FMT> is ignored for literals, including \$prop.xxx variables.

The following table describes these tags:

Property	Description
----------	-------------

Property	Description																						
ESC_STYLE= NONE C HTML JS JS2 URL	<p>Specifies the escape type of the formatted text. Valid values are:</p> <p>NONE Default setting. Specifies that no special treatment be given to any character in the content body.</p> <p>C Give special treatment to the characters ', ", \, \r, ` , and \n, which are meaningful in C programs. These characters will be escaped.</p> <p>HTML Give special treatment to the following characters, which are meaningful in HTML text:</p> <table><tr><td>&</td><td>becomes &amp;</td></tr><tr><td>'</td><td>becomes &apos;</td></tr><tr><td>"</td><td>becomes &quot;</td></tr><tr><td><</td><td>becomes &lt;</td></tr><tr><td>></td><td>becomes %gt;</td></tr></table> <p>JS Give special treatment to the following characters, which are meaningful in JavaScript text:</p> <table><tr><td>'</td><td>becomes %27</td></tr><tr><td>"</td><td>becomes %22</td></tr><tr><td>/</td><td>becomes %2F</td></tr><tr><td>\</td><td>becomes %5C</td></tr><tr><td>\r</td><td>becomes %0D</td></tr><tr><td>\n</td><td>becomes %0A</td></tr></table> <p>JS2 Same as JS, but give no special treatment to the character, /, and give special treatment to two additional characters:</p> <ul style="list-style-type: none">- % becomes %25- Line breaks are suffixed with %0A <p>URL Translate all characters other than letters, digits, and '@*-_#' to '%xx', where xx is the hexadecimal coding of the translated character.</p>	&	becomes &	'	becomes '	"	becomes "	<	becomes <	>	becomes %gt;	'	becomes %27	"	becomes %22	/	becomes %2F	\	becomes %5C	\r	becomes %0D	\n	becomes %0A
&	becomes &																						
'	becomes '																						
"	becomes "																						
<	becomes <																						
>	becomes %gt;																						
'	becomes %27																						
"	becomes %22																						
/	becomes %2F																						
\	becomes %5C																						
\r	becomes %0D																						
\n	becomes %0A																						

Property	Description
JUSTIFY=LEFT CENTER RIGHT TRUNCATE WRAP LINE	<p>Specifies the justification of the formatted text. Valid values include:</p> <p>TRUNCATE Default setting. Eliminates HTML formatting by replacing '<' and '>' with &lt; and &gt; (but see KEEPLINKS and KEEPTAGS).</p> <p>LEFT CENTER RIGHT Produces exactly WIDTH characters, truncated or padded with spaces as necessary, with any embedded new lines replaced by a single space, and the output text delimited by <pre> and </pre> tags. The WIDTH argument must be specified as a positive integer.</p> <p>WRAP Same as LEFT, except that text wrapping honors word boundaries (line breaks are not placed within words).</p> <p>LINE Same as TRUNCATE, except that it also replaces all embedded line breaks with
 tags.</p>
KEEPLINKS=YES NO	If KEEPLINKS=YES is specified, the action of JUSTIFY=LINE or JUSTIFY=TRUNCATE is modified to preserve HTML anchor tags (Action:) while converting all other '<' and '>' characters. Mutually exclusive with KEEPTAGS.
KEEPNL=YES NO	The normal action of PDM_FMT is to convert all embedded newlines and any following spaces to a single space. If KEEPNL=YES is specified, embedded newlines are preserved. This argument is ignored for JUSTIFY=LINE.
KEEPTAGS=YES NO	If KEEPTAGS=YES is specified, the action of JUSTIFY=LINE or JUSTIFY=TRUNCATE is modified to preserve all HTML tags. Mutually exclusive with KEEPLINKS.
PAD=YES NO	If PAD=NO is specified, PDM_FMT does not convert empty strings to a single space. This is the normal action when WIDTH is non-zero, or JUSTIFY is TRUNCATE or WRAP.
WIDTH= <i>nn</i>	When non-zero, specifies that the text should be formatted to exactly WIDTH characters.

<PDM_FMT> without WIDTH or JUSTIFY does no formatting on the enclosed text, but surrounds the text with <pre>and </pre>.

For example, to produce a multi-line description, enter the following:

```
<PDM_FMT WIDTH=50 JUSTIFY=WRAP>$args.description</PDM_FMT>
```

To produce multi-column output, enter the following:

```
<PDM_FMT><PDM_FMT WIDTH=20 JUSTIFY=LEFT>$cst.last_name</PDM_FMT>
<PDM_FMT WIDTH=20 JUSTIFY=LEFT>$cst.first_name</PDM_FMT>
<PDM_FMT WIDTH=20 JUSTIFY=TRUNCATE>$cst.middle_name</PDM_FMT>
</PDM_FMT>
```

PDM_IF: Conditional Processing

These tags are used to conditionally include text. <PDM_IF> blocks can be placed anywhere in an HTMPL file - in HTML, in JavaScript, and even within HTML tags. <PDM_IF> and <PDM_ELIF> (else if) both take a simple conditional clause as their properties rather than name-value pairs. If the clause is true, the text after the tag to the closing tag is included in the file; if the clause is false, the server discards the text between the tag and the closing tag. The closing tag can be <PDM_ELIF>, <PDM_ELSE>, or </PDM_IF>.

The <PDM_ELSE> and <PDM_ELIF> tags are optional. If both are specified, all <PDM_ELIF> tags must precede <PDM_ELSE>. There can be any number of <PDM_ELIF> tags between <PDM_IF> and <PDM_ELSE> (or </PDM_IF> if <PDM_ELSE> is omitted).

The syntax of the conditional in <PDM_IF> and <PDM_ELIF> is as follows:

- 0 is false; any other number is true
- “” is false; “*any-string*” is true
- “*value op value*” evaluates the left and right values against each other according to *op*. If both values consist of digits (optionally preceded by - or +), the comparisons are done numerically. Otherwise, they are done lexically (ASCII collation). Valid *op* values include:

op Value	Description
==	Equal to
!=	Not equal to
>=	Equal to or greater than (must be written as \>= or >=)

op Value	Description
<	Less than (must be written as \< or <)
>	Greater than (must be written as \> or >)
<=	Equal to or less than (must be written as \<= or <=)
&	Performs a bit-and of the left and right values. True if any bits are set; false if none are set.
%	Returns true if the left value is an even multiple of the right value, and false otherwise (useful for building two-dimensional tables).
:	Performs a byte-oriented pattern match like the UNIX grep command. It returns true if the left value contains the regular expression defined by the right value.

For example:

```
<PDM_IF $count \>= 10> . . .
<PDM_ELIF $count < 5> . . .
<PDM_ELSE> . . .
</PDM_IF>
```

There can be more than one conditional in a PDM_IF statement. Conditionals are separated by connectors, either && (and) or || (or). There is no precedence for either connector. The web engine examines a conditional from left to right until it reaches a connector. If the initial condition is true and the connector is ||, it considers the entire condition to be true without further evaluation. If the initial condition is false and the connector is &&, it considers the entire condition to be false without further evaluation. Otherwise, it considers the condition undetermined, and evaluates the conditional from after the connector.

PDM_INCLUDE: Inserting from a Different File

The <PDM_INCLUDE> tag is used to insert text from a second file into an HTMPL file. The server replaces the <PDM_INCLUDE> tag with the contents of the second file.

Included files can themselves contain <PDM_INCLUDE> tags. There is no limit to the depth of nesting.

The <PDM_INCLUDE> tag supports the following properties:

Property	Description
FILE=filename	(Required) Specifies the file to include. The web engine searches the directories used for HTMPL files, as defined in the current user's access type.

Property	Description
FIXUP=[YES N O]	<p>(Optional) Indicates whether the file should be interpreted by the web interface like a normal HTML template file, such as expanding variables beginning with dollar signs (\$) and interpreting other Unicenter Service Desk tags, such as PDM_LIST, and PDM_FORMAT. The value YES, indicates that the file should be treated as a regular HTML template file, and the value NO means that the included file should be treated as literal text. The default is YES.</p> <p>Note: For compatibility with previous releases, the values TRUE or 1 can be substituted for YES, and the values FALSE or 0 can be substituted for NO. These values are deprecated, and should not be used in new pages.</p>
propname=value	<p>Specifies that property propname should have the specified value. The property's value can be accessed within the included file by prefixing propname with \$prop. For example, the following specification would allow the included file to reference \$prop.menubar:</p> <pre><PDM_INCLUDE ... menubar=no></pre> <p>Global properties can also be specified in the web.cfg configuration file (for more information on web.cfg, see Configuration File Properties in the "Configure the Web Interface" chapter of the Administrator Guide).</p> <p>Note: For compatibility with previous releases, property values specified on <PDM_INCLUDE> can be referenced without the preceding 'prop.', in the form \$propname. This usage is deprecated and should not be used in new pages.</p>

PDM_JSCRIPT: Conditionally Include a JavaScript File

The <PDM_JSCRIPT> tag is used to conditionally include a JavaScript file on a form. This tag has two forms:

```
<PDM_JSCRIPT file=xxxx.js [include=yes|no]>
```

Pdm_jscript with file=xxx.js specifies that JavaScript file xxx.js is required by this form. The webengine adds the file to a list of JavaScript files required by the form. Processing of the tag occurs while the form is being parsed, and is not affected by pdm_if. That is, a pdm_jscript tag referencing a file adds that file to the list of JavaScript files if it occurs anywhere in the file or in an included file, or in a macro.

The optional argument include=no can be specified to tell the webengine to ignore the tag. This argument provides conditional processing for the tag, and is primarily useful when the tag is invoked in a macro. For example, the dtlTextbox macro specifies:

```
<PDM_JSCRIPT file=spellcheck.js include=&{spellchk}>
```

This indicates that any form containing a dtlTextbox macro that specifies spellchk=yes requires the JavaScript file spellcheck.js.

The second form of the pdm_jscript tag is:

```
<PDM_JSCRIPT insert=here>
```

Pdm_jscript with insert=here asks the webengine to insert standard HTML <script> tags for all required JavaScript files. The webengine processes this form of the tag during the HTML generation phase, so that it is affected by pdm_if. A pdm_jscript tag with insert=here is part of std_head_include.htmpl, so it is present on virtually every form.

Note: The webengine inserts script tags only the first time it encounters pdm_jscript insert=here.

PDM_LINK: Create a Hyperlink Invoking an HTMPL Operation

<PDM_LINK> and </PDM_LINK> can be added to any web interface HTML template to create links a link that invokes an HTMPL operation. The <PDM_LINK> tag generates the standard HTML tag and has similar arguments, except that it allows specification of a Unicenter Service Desk operation in place of a URL.

The format is as follows, where *operation* is one of the supported operations (see page 163):

```
<PDM_LINK OP=operation> ... </PDM_LINK>
```

For example:

```
<PDM_LINK OP=MENU> Menu </PDM_LINK>
<PDM_LINK OP=CREATE_NEW FACTORY=iss> Submit Issue </PDM_LINK>
<PDM_LINK OP=LOGOUT> Logout </PDM_LINK>
```

PDM_LIST: Format a List of Database Rows

The <PDM_LIST> and </PDM_LIST> tags are used to delimit repeating sections of HTML for multi-record output. Everything between <PDM_LIST> and </PDM_LIST> is repeated once for each record to be output. There are two types of PDM_LISTs:

- Lists taken from an object attribute that implies a list. For example, the properties attribute of the request object is the list of properties associated with that request. This type of PDM_LIST always has a SOURCE property.
- Lists with an explicit where clause. This type of PDM_LIST always has a WHERE property.

An object attribute <PDM_LIST> takes the following properties:

Property	Description
----------	-------------

Property	Description																						
ESC_STYLE= NONE C HTML JS JS2 URL	<p>Specifies the escape type of the formatted text. Valid values are:</p> <p>NONE Default setting. Specifies that no special treatment be given to any character in the content body.</p> <p>C Give special treatment to the characters ', ", \, \r, ', and \n, which are meaningful in C programs. These characters will be escaped.</p> <p>HTML Give special treatment to the following characters, which are meaningful in HTML text:</p> <table> <tbody> <tr><td>&</td><td>becomes &amp;</td></tr> <tr><td>'</td><td>becomes &apos;</td></tr> <tr><td>"</td><td>becomes &quot;</td></tr> <tr><td><</td><td>becomes &lt;</td></tr> <tr><td>></td><td>becomes %gt;</td></tr> </tbody> </table> <p>JS Give special treatment to the following characters, which are meaningful in JavaScript text:</p> <table> <tbody> <tr><td>'</td><td>becomes %27</td></tr> <tr><td>"</td><td>becomes %22</td></tr> <tr><td>/</td><td>becomes %2F</td></tr> <tr><td>\</td><td>becomes %5C</td></tr> <tr><td>\r</td><td>becomes %0D</td></tr> <tr><td>\n</td><td>becomes %0A</td></tr> </tbody> </table> <p>JS2 Same as JS, but give no special treatment to the character, /, and give special treatment to two additional characters:</p> <ul style="list-style-type: none"> - % becomes %25 - Line breaks are suffixed with %0A <p>URL Translate all characters other than letters, digits, and '@*-_#' to '%xx', where xx is the hexadecimal coding of the translated character.</p>	&	becomes &	'	becomes '	"	becomes "	<	becomes <	>	becomes %gt;	'	becomes %27	"	becomes %22	/	becomes %2F	\	becomes %5C	\r	becomes %0D	\n	becomes %0A
&	becomes &																						
'	becomes '																						
"	becomes "																						
<	becomes <																						
>	becomes %gt;																						
'	becomes %27																						
"	becomes %22																						
/	becomes %2F																						
\	becomes %5C																						
\r	becomes %0D																						
\n	becomes %0A																						
LENGTH=nn	Specifies the number of rows of output (defaults to all).																						

Property	Description
PREFIX= <i>prefix</i>	Specifies the prefix on references to attributes from records in the list. These are referenced in the form <code>\$prefix.attr_name</code> in the text between <code><PDM_LIST></code> and <code></PDM_LIST></code> . The PREFIX property is optional in an object variable list. If PREFIX is omitted, the value of SOURCE is also used for the prefix.
SEARCH_TYPE=DISPL AY	Specifies the method the server should use to build the list form:
GET_DOB	<p>DISPLAY specifies the server should issue a single query for the entire form</p> <p>GET_DOB specifies the server should issue separate queries for each row of the form</p> <p>The choice affects list performance, and depends on the complexity of the list (the number of joins required to display it) and the characteristics of your DBMS. GET_DOB has more predictable performance than DISPLAY, and is the default.</p>
SORT= <i>index-name</i>	Specifies the index name to use for sorting. The default value of this argument is DEFAULT (which means the first sort index for the underlying factory).
SOURCE= <i>source</i>	Specifies the object variable defining this list. This field is required. Do not put a dollar sign (\$) in front of <i>source</i> on the PDM_LIST statement itself. If the PREFIX property is not specified, <i>source</i> is also used as the prefix for references to attributes from records on the list, in references of the form <code>\$source.attr_name</code> . When used in a reference, <i>source</i> does require a preceding dollar sign.
START= <i>nn</i>	Specifies the first output row (defaults to zero).

For example:

```
<table border>
<tr>
<th>Child Change Order Number</th>
<th>Summary</th>
</tr>
<PDM_LIST SOURCE=args.children>
<tr>
<td>$args.children.chg_ref_num</td>
<td>$args.children.summary</td>
</tr>
</PDM_LIST>
</table>
```

Because no prefix was specified, references to attributes of the listed records are prefixed by \$args.children, the source value.

A where clause PDM_LIST takes the following properties:

Property	Description
FACTORY= <i>name</i>	Specifies a class of object to be searched. This property is required.
LENGTH= <i>nn</i>	Specifies the number of rows of output (defaults to all).
ORDER_BY= <i>attr-name</i>	Specify the attribute name to sort by. It can contain the DESC (descending) or ASC (ascending) modifiers.
PREFIX= <i>prefix</i>	Specifies the prefix on references to attributes from records in the list. These are referenced in the form \$ <i>prefix.attr_name</i> in the text between <PDM_LIST> and </PDM_LIST>. The PREFIX property is required in a where clause list.
START= <i>nn</i>	Specifies the first output row (defaults to zero).
WHERE= <i>where-clause</i>	Specify the where clause for the search. It can contain (dotted) attributes. This property is required.

For example:

```
<table>
<tr>
<th>Child Change Order Number</th>
<th>Summary</th>
</tr>
<PDM_LIST PREFIX=list FACTORY=chg WHERE="status = 'OP'">
<tr>
<td>$list.chg_ref_num</td>
<td>$list.summary</td>
</tr>
</PDM_LIST>
</table>
```

PDM_MACRO: Insert Text from a Macro File

The PDM_MACRO Tag

The <PDM_MACRO> tag is used to insert a macro file into an HTMPL file. Its functionality is similar to PDM_INCLUDE, with two important differences:

- A file included by PDM_MACRO has a formal argument list, with required arguments and arguments with default values.
- A file included by PDM_MACRO always comes from the directory specified for the configuration property MacroPath, regardless of the current user's access type.

NAME=macroname

(Required) Specifies the macro to include. The web engine affixes the suffix ".mac" and searches for the file in the path specified by configuration file property MacroPath.

Other properties may be required, depending on the macro included. A macro file has the general layout:

```
comments
#args
name1 [= value1]
name2 [= value2]
...
#data
data to insert
```

The following descriptions explains the file layout, line by line:

comments

The only valid statements in a macro prior to the #args statement are comments. Comments are indicated by either a # sign or a // as their first non-blank character or characters.

#args

Must be coded exactly as shown, with the # sign in column one and no other information on the line. This statement begins the args section, which can contain argument definitions and comments.

name [= value]

Defines an argument for the macro. Only arguments explicitly mentioned in the args section are valid for the macro. A value specified for an argument in the args section is that argument's default value. Arguments without a default value are required, and must be supplied by the caller on the <PDM_MACRO> statement itself.

#data

Must be coded exactly as shown, with the # sign in column one and no other information on the line. This statement begins the data section, which is the part of the macro inserted into the file using PDM_MACRO. Everything in the data section is inserted into the calling file, including lines that would be comments prior to the data section.

data to insert

The data to insert into the calling file. This data can contain references to arguments in the form:

&{ *arg_name* }

These references are replaced with the value of the argument supplied by the caller, or with the default value if the caller did not supply a value.

The web engine normally reads a macro file only once, the first time it is used, and then stores the parsed macro in its own memory. This improves performance, but can be inconvenient if you are developing a macro. Use the configuration file property SuppressMacroCache to prevent this behavior and cause the web engine to discard all macros in its memory each time it begins processing a new form.

To Comment Out PDM_MACRO Tags

To comment out <PDM_MACRO> tags, enter an exclamation point in front of the P as follows: <!PDM_MACRO>. To prevent the browser from processing the commented out portion of the form, place <PDM_IF 0> before the <!PDM_MACRO> tag, and </PDM_IF> after the line you commented out.

Example:

```
<PDM_IF 0>

<!PDM_MACRO NAME=dtlDropdown hdr="Status" attr=status lookup=no
evt="onBlur=\\"\\\"detailSyncEditForms(this)\\\"\\\">
<!PDM_MACRO NAME=dtlDropdown hdr="Priority" attr=priority lookup=no
evt="onBlur=\\"\\\"detailSyncEditForms(this)\\\"\\\">

</PDM_IF>
```

Predefined Macros Used by Web Screen Painter

There are a number of predefined macros included with Unicenter Service Desk. The majority of these insert JavaScript text to create an element on a web form. Use Web Screen Painter to create and modify forms using these macros. See the *Web Screen Painter* online help for information on their arguments.

Detail Form Macros

button

Insert a graphic button.

dtlCheckbox

Insert a check box on a detail form.

dtlDate

Insert a date field on a detail form.

dtlDateReadonly

Insert a read-only date field on a detail form.

dtlDropdown

Insert a drop-down select box on a detail form.

dtlEnd

End a detail form.

dtlEndTable

End a table within a detail form.

dtlForm

Begin a detail form.

dtlHTMLEditBox

Insert a detail form field that is a text box containing an HTML editor.

dtlHier

Insert a detail form field that is a text box validated against an external table with a hierarchical lookup.

dtlLookup

Insert a detail form field that is a text box validated against an external table.

dtlLookupReadonly

Insert a detail form field that is a read-only hyperlink to an external table.

dtlReadonly

Insert a read-only text field on a detail form.

dtlStart

Begin the first table in a detail form.

dtlStartExpRow

Begin an expandable row on a detail form.

dtIStartRow

Begin a normal row on a detail form.

dtITextbox

Insert a text box on a detail form.

List Form Macros

lsCol

Specify a column in a list form.

lsEnd

End the list portion of a list form.

lsStart

Begin the list portion of a list form.

lsWrite

Insert text into the repeating section of a list form.

sfDate

Insert a date field in a search filter.

sfDropdown

Insert a drop-down select box on a search filter.

sfEnd

End a search filter.

sfHier

Insert a search filter field that is a text box validated against an external table with a hierarchical lookup.

sfLookup

Insert a search filter field that is a text box validated against an external table.

sfStart

Begin a search filter.

sfStartRow

Begin a row within a search filter.

sfTextbox

Insert a textbox into a search filter.

Menubar Macros

endMenu

End a menu within a menu bar.

menuItem

Define a global item on a menu.

endMenubar

End a menu bar.

menuItemLocal

Define an item on a menu invoked in the context of the current window.

menubarItem

Define a menu within a menu bar.

startMenu

Begin a menu within a menu bar.

startMenubar

Start a menu bar.

PDM_NOTEBOOK: Create a Notebook

Several of the forms in the Unicenter Service Desk analyst interface use a notebook control. A notebook allows several sets of fields to be displayed in the same physical area of the screen, with only one set visible at a time. The user selects the set of fields that is visible by clicking a named tab at the top of the notebook, or by pressing the access key combination Alt+n, where n is the number of the tab. An example of a form using a notebook is the Issue Detail (detail_iss.htmpl). We recommend that you use Web Screen Painter to modify the contents of notebooks, or insert a notebook into a form that does not already contain one. See *Web Screen Painter* online help for more on notebooks and notebook tabs.

The </PDM_NOTEBOOK> tag marks the end of a notebook. We recommend that you provide this tag in order to ensure compatibility with future releases. However, in this release it produces no output code and is optional.

PDM_PRAGMA: Specify Server Information

The <PDM_PRAGMA> tag is used to specify information used by the web engine, such as form release and version. It does not generate any HTML code, and can be placed anywhere in a form. Possible arguments are:

Argument	Description
RELEASE=value	Specifies the Unicenter Service Desk release number corresponding to this form. This value is "110" on all r11 forms. It is accessible within the form in the \$prop.release variable.
SITEMOD=value	Specifies a site-defined string identifying the modifications applied to this form. It is accessible within the form in the \$prop.sitemod variable.
VERSION=value	Specifies a CA-defined string identifying the version number of this form. It is accessible within the form in the \$prop.version variable.
OVERIDE=YES NO	Specifies whether or not values in this PDM_PRAGMA statement override values in previous PDM_PRAGMA statements.

CA uses PDM_PRAGMA statements to document form versions. All r11 forms include the following PDM_PRAGMA statement:

```
<PDM_PRAGMA RELEASE=110>
```

In addition, the std_head.htmpl form includes the following JavaScript statement:

```
cfgFormRelease = "$prop.release" - 0;
```

The PDM_PRAGMA statement and the cfgFormRelease variable allows the Unicenter Service Desk web interface to distinguish r11 forms from previous release forms. Releases prior to r6.0 did not support the PDM_PRAGMA statement.

Normally, only PDM_PRAGMA statements in the highest-level file of a form (that is, a file not brought in by PDM_INCLUDE) are used to set \$prop.release, \$prop.sitemod, and \$prop.version. In addition, a PDM_PRAGMA statement will not override a non-empty value set by a previous PDM_PRAGMA statement. You can specify OVERRIDE=YES to specify that a PDM_PRAGMA statement can override previous PDM_PRAGMA statements, or that a PDM_PRAGMA statement in an included file can be used.

PDM_SCOREBOARD: Build a Scoreboard Tree

The <PDM_SCOREBOARD> tag is used to generate the scoreboard shown on the left side of the main form. It takes the following property:

TARGET=*value*

Specifies the name of the target frame for lists requested by clicking a node on the scoreboard. Lists are loaded into the target specified, which can be any value supported for the target attribute of a link. The default value is **_self** (the window containing the PDM_SCOREBOARD tag).

Any HTMPL form including a <PDM_SCOREBOARD> tag must also include the fldrtree.js JavaScript file. This can be included with the following statement in <HEAD> section of the form:

```
<SCRIPT LANGUAGE="JavaScript" SRC="$CAisd/CAisd/fldrtee.js"></SCRIPT>
```

In addition, it is desirable to include a link with the name `scoreboard_asof_data` to display the effective date of the numbers in the tree. See the distributed file `scoreboard.htmpl` for an example of the use of this tag.

The queries included on the scoreboard are defined by the contents of the User_Query table (object name usq) for the current user. A record in this table defines each line on the tree (folder or node).

Initially, users have no entries in their User_Query table. A user with no User_Query entries receives the default set of scoreboard queries associated with their access type. A user with administrative authority can also customize the default scoreboard for an access type.

PDM_SET: Set the Value of a Server Variable

The <PDM_SET> tag is used to assign a value to a server variable. It has the following syntax:

<PDM_SET arg.name[+] = value>

arg

(Required) Specifies the variable type, and must be arg for normal use.

Note: There is no \$ character.

Name

(Required) Specifies the name of the variable.

+

(Optional) Specifies that the value should be appended to the existing value of the variable. There cannot be any spaces before or after.

=

(Required) Must be specified exactly as shown, with no spaces before or after.

value

(Required) Specifies the text to be assigned or appended to the variable.

The PDM_SET tag can also be used in the preprocessor phase to create or update a preprocessor variable. This is an advanced use, discussed in Web Engine PreProcessing (see page 183).

PDM_TAB: Create a Tab within a Notebook

The <PDM_TAB> tag is used to define a notebook tab. We recommend that you use Web Screen Painter to modify the contents of notebooks, or insert a notebook into a form that does not already contain one. See *Web Screen Painter* online help for more on notebooks and notebook tabs.

PDM_WSP: Control WSP Preview

The <PDM_WSP> tag is used to control the Web Screen Painter preview feature. It does not generate any HTML code, and can be placed anywhere in a form.

By default, WSP determines how to preview a form by examining the form name:

- For detail forms (names of the form `detail_factory.htmpl`), WSP displays the form in edit view, with data from the most recently created row of the appropriate table. If there is no data you are allowed to view in the table, WSP displays the form set up to create a new row. Note that WSP preview sessions are normally prohibited from updating the database. WSP displays forms in edit view to allow you to preview all features. However, Unicenter Service Desk ignores a Save request from a read-only preview session. The web engine changes the text on the Save button to `noSave` as a visual reminder of this.
- For list forms (names of the form `list_factory.htmpl`), WSP displays the form in list view, with the list showing data from the most recently created row of the appropriate table. If there is no data you are allowed to view in the table, WSP displays the form in search view, with the filter open.
- For other forms, WSP simply displays the form with no database context.

You can change this default behavior by placing a PDM_WSP tag anywhere on the form. For example, you can display a notebook tab form on its associated detail form, or provide prerequisite arguments for forms normally invoked with an environment provided by another form. Possible arguments are:

Property	Description
<code>FACTORY=</code> <i>value</i>	Specifies the Object Engine factory used by this form.
<code>PREVIEW=</code> <i>name.htmpl</i> <i>value</i> <code>no</code>	Specifies the preview URL. This can be an HTMPL file name, in the form <code>xxxx.htmpl</code> ; a Unicenter Service Desk URL (used unaltered if it begins with <code>"OP=</code> "); or the keyword <code>"no"</code> , indicating the form cannot be previewed. A value not beginning <code>OP=</code> is modified by replacing a reference of the form <code>{factory}</code> or <code>{factory:}</code> with an ID or persistent ID (respectively) of the most-recently created row from the referenced factory that the current user is authorized to view.
<code>WHERE=</code> <i>value</i>	Specifies a where clause used to search for a representative row or rows to show on the previewed form.

Server Variables

Unicenter Service Desk information is included in the HTML template using variables beginning with a dollar sign (\$). Each page is created with some variables that are documented in the template file. These variables can be put on the page or used in conditional statements:

- Simple Variables
- Property Variables
- Environment Variables
- Business Object Variables
- List Variables

Simple Variables

Simple variables specify flags that are passed to the web page. To access a simple variable, use the variable name preceded by a dollar sign (\$). This makes the value of the variable available. For example, two such variables are \$CAisd and \$cgi. Putting \$CAisd in a template results in the substitution of the main Unicenter Service Desk web server installation directory, whereas \$cgi refers to the URL of the pdmweb.exe program. Simple variables are documented in the upper section of the HTMPL file that uses them.

The following shows a list of variables that can be used in all the HTMPL files:

\$ACCESS.group

The user access privilege object contains the privilege settings on the function group *group* for the current login user. For example, \$ACCESS.admin holds the privilege value for the admin functional group. Valid privilege values are:

- 0-NO ACCESS
- 1-VIEW
- 2-MODIFY

This variable is not available in the login form.

\$cgi

The URL of the pdmweb.exe program.

\$cst

The data object of the current login user. This variable is not available in the login form. You can reference individual attributes of this object with the form \$cst.*attrname*; for example, \$cst.first_name.

\$CAisd

The URL of main Unicenter Service Desk web server installation directory.

\$MachineName

The MachineName defined in the web.cfg file (for more information on web.cfg, see Configuration File Properties in the “Configure the Web Interface” chapter of the *Administrator Guide*).

\$ProductName

The product name defined in the NX.env file.

\$SESSION

The session object saves all session variables including session ID (\$SESSION.SID) and all variables defined in the web.cfg file (for more information on web.cfg, see Configuration File Properties in the “Configure the Web Interface” chapter of the *Administrator Guide*).

\$USER_STATE

User-defined state information.

Property Variables

Property variables represent a property of the configuration file, web.cfg. You can access any entry in the web.cfg file (including user-defined entries) within an HTML template file by prefixing it with “\$prop.”

For example, one of the lines in web.cfg, which specifies the number of entries displayed in a single page on a list form is as follows:

```
ListPageLength 10
```

You can refer to this variable in an HTML template with the specification:

```
$prop.ListPageLength
```

If you use the <PDM_INCLUDE> special tag to incorporate another file into a template file, you can specify additional properties as attributes of the <PDM_INCLUDE> tag. You can reference these properties in the included file in the same way as web.cfg properties. A property specified as a <PDM_INCLUDE> attribute that has the same name as a web.cfg property overrides the web.cfg property within the included file.

For example, the following <PDM_INCLUDE> tag creates a property called \$prop.menubar that can be referenced within the std_body.htmpl file:

```
<PDM_INCLUDE FILE=std_body.htmpl menubar=no>
```

Note: You can refer to configuration file property *xxx* in two ways: `$prop.xxx` or `$SESSION.xxx`. Both return the same value. However, the `$prop.xxx` syntax is preferred because it involves less server overhead.

In addition to properties from web.cfg, there are several predefined properties that can be accessed with `$prop`. These are:

`$prop.browser`

A string identifying the browser in use. This will be "IE" for Internet Explorer.

`$prop.combo_name`

A string containing the current user's name, in the form "last_name, first_name middle_name."

`$prop.factory`

A string containing the factory associated with the current form, such as "cr" for requests or "iss" for issues.

`$prop.FID`

A string containing the numeric form ID of the current form.

`$prop.form_name`

A string containing the name of the current HTML template, in the form *xxx.htmpl*.

`$prop.form_name_1`

A string containing the substring of the form name before the first underscore. For example, for the form *detail_chg_edit.htmpl*, `form_name_1` would be "detail."

`$prop.form_name_2`

A string containing the substring of the form name after the first underscore and before the last underscore (or dot). For example, for the form *detail_chg_edit.htmpl*, `form_name_2` would be "chg."

`$prop.form_name_3`

A string containing the substring of the form name after the last underscore and before the dot. For example, for the form *detail_chg_edit.htmpl*, `form_name_3` would be "edit." For the combination detail form, which has a file name of the form *detail_xxx.htmpl*, `$prop.form_name_3` is set to the current view, either "ro" or "edit".

`$prop.release`

A string containing the release level of the form. The PDM_PRAGMA (see page 152) statement in HTML Tags chapter contains more details on this property.

\$prop.SID

A string containing the numeric session ID of the current session.

\$prop.sitemod

A string containing the site-defined modification name of the form. The (see page 152)PDM_PRAGMA statement in HTML Tags chapter contains more details on this property.

\$prop.user_type

A string containing "analyst," "customer," "employee," or "guest."

\$prop.version

A string containing the version of the form. The PDM_PRAGMA (see page 152) statement in HTML Tags chapter contains more details on this property.

Environment Variables

Environment variables represent an entry within the NX.env configuration file. You can reference any entry in NX.env within an HTMPL template file by prefixing it with "\$env."

For example, one of the lines in NX.env, which specifies the host name of the Unicenter Service Desk server is as follows:

```
@NX_SERVER=hostname
```

You can refer to this variable within an HTMPL template file with the specification:

```
$env.NX_SERVER
```

Business Object Variables

Business object variables represent a Unicenter Service Desk object, such as an issue or a request. To access an object, you need to start with the variable name, followed by a period (.), followed by whatever attribute names you want to display. For example, on an issue where, by convention, the object is represented by the variable args, you can display the description, the open date, the assignee's phone number, the number of activities on the issue, and the description of the first activity, as shown by the following:

```
$args.description  
$args.open_date  
$args.assignee.phone_number  
$args.act_log.length  
$args.act_log.0.description
```

You can use braces to delimit the variable name if it is not surrounded by white space. For example, \$foo bar and \${foo}bar are both valid. You can also use the variable args to access non-attribute values (for example, \$args.KEEP.name as described in Supported Operations (see page 163)).

It is possible that a non-attribute variable may not be defined. For example, it may be possible to get to a form from two different places, only one of which provides a value for \$args.KEEP.foo. You can provide a default value for a \$args reference with the following syntax, where the string after the colon is substituted for the reference if *variable* is undefined:

```
 ${args.variable:default}
```

Time Zone Date Variables

Time zone date variables are a special case of business object variables. They provide a means to convert universal dates (UTC) represented as integers to string dates adjusted for the time zone of the user's browser. The variable for representing integer dates is:

```
$args.attr_name_INT_DATE
```

For example: \$args.open_date_INT_DATE

Factory Data Variables

Factory data variables are a special case of business object variables. A factory data variable is replaced by information about a referenced object. There are seven such variables available:

\$args.attr_name.COMMON_NAME

The common name (externally readable string) of the table referenced by the attribute. For example, on the Request Detail form, the value of \$args.assignee.COMMON_NAME is the assignee's combo name ("last, first, middle").

\$args.attr_name.COMMON_NAME_ATTR

The attribute name of the common name in the table referenced by the attribute. For example, on the Request Detail form, the value of \$args.assignee.COMMON_NAME_ATTR is "combo_name".

\$args.FACTORY_attr_name

The name of the factory associated with the specified attribute. For example, on the Request Detail form, the value of \$args.FACTORY_assignee is "agt".

\$args.LENGTH_attr_name

The maximum length of the attribute. For example, on the Request Detail form, the value of \$args.LENGTH_summary is 240.

\$args.attr_name.REL_ATTR

The rel attr (foreign key) of the attribute. For example, on the Request Detail form, the value of \$args.assignee.REL_ATTR is the value of the assignee's ID field.

\$args.attr_name.REL_ATTR_ATTR

The attribute name of the rel_attr in the table referenced by the attribute. For example, on the Request Detail form, the value of \$args.assignee.REL_ATTR_ATTR is "id".

\$args.REQUIRED_attr_name

A string, either "0" or "1" indicating whether the referenced attribute is required.

\$args.attr_name.SELECTIONS

A list of valid selections for *attr_name*. This value is an empty string if *attr_name* is not a reference to another table, or if the size of table referenced by *attr_name* exceeds the value of the configuration file property SelListCacheMax. Otherwise, the SELECTIONS variable is a string containing the common name and rel attr of all the entries in the referenced table. Successive values are separated by the string "@,@", so the variable's value has the form:

"cname1@,@rel_attr1@,@cname2@,@rel_attr2"

\$args.factory_SEL_UNDER_LIMIT

A string, either "0" or "1", indicating whether the current number of rows in the table corresponding to *factory* is less than the value of the configuration file property SelListCacheMax. This variable is deprecated in favor of the SELECTIONS variable, which should be used in all new forms.

Factory data variables containing a dotted reference (COMMON_NAME, REL_ATTR, and SELECTIONS) can be used with a dotted reference of any length. For example, on a Request Detail form
\$args.assignee.organization.COMMON_NAME is replaced by the external name of the assignee's organization.

List Variables

List variables are used to iterate through data. They are accessed using list tags as described in PDM_LIST: Format a List of Database Rows (see page 142).

Server Operations

Supported Operations

The following operations are supported to let you integrate the Unicenter Service Desk web pages with your web pages:

CREATE_NEW

Provides a generic interface to allow the user to create a new row in a specified table. The object name must be specified, and by default a template named *detail_xxx_edit.htmpl* is used for object *xxx*. You can override the *.htmpl* file by specifying the HTMPL property.

Required specifiers:

FACTORY=*object-name*

Optional specifiers:

ALG_PRESET=*preset_expression*
ALG_PRESET_REL=*preset_expression*
CREATE_ALG=*activity_log_type*
HTMPL=zdetailxxx_factory.htmpl
KEEP.attr_name=*value*
PRESET=*preset_expression*
PRESET_REL=*preset expression*
SET.attr_name=*value*
use_template=1 | 0 (0 is the default)

Note: In order to use the HTMPL specifier with CREATE_NEW, the referenced form must have a name conforming to the naming convention shown above. It must begin with the string "zdetail", followed by any alphanumeric characters (including a null string), followed by an underscore and the factory name.

ENDSESSION or LOGOUT

Ends the current logged-in session. ENDSSESSION is the preferred operation.

GENERIC_LIST

Provides a generic interface to allow the user to display a list from any table in the database. The object name must be specified, and by default a template named *list_xxx.htmpl* is used for object *xxx*. You can override the *.htmpl* file by specifying the HTMPL property.

Required specifiers:

FACTORY=*object-name*
KEEP.attr_name=*value*

JUST_GRONK_IT

Provides a generic interface to allow the user to display any customized form.

Required specifiers:

HTMPL=*htmpl_file*

MENU

Displays the main menu screen, which is defined in the web.cfg file in the Menu property (see Configuration File Properties in the “Configure the Web Interface” chapter of the *Administrator Guide* for more information on web.cfg).

Optional specifiers:

HTMPL=*menufile*

menufile is the name of an alternate main menu file.

PAGE_EXTENSION

Allows the webmaster to specify additional extensions to the interface.

Required specifiers:

NAME=*html_file*

html_file is one of the file names listed in the configuration file UserPageExtensions directive (see Configure File Properties in the Configure the Web Interface chapter of the *Administrator Guide* for more information on web.cfg).

Optional specifiers:

REQUIRES_LOGIN=1

If present, a login screen appears first if the user is not currently logged in. If omitted or set to zero, the file is shown without checking if the user is currently logged in.

RELOG

Displays the login screen.

SEARCH

Provides a generic interface to allow the searching of any table in the database. This assumes that an appropriate search_*xxx.htmpl* has been created, where *xxx* is the *object-name*, as defined in the various .maj files in the majic directory in bopcfg (see the Objects and Attributes appendix for more information). By default, the results of this search are displayed in list_*xxx.htmpl*, but this may be overridden by specifying the HTMPL property.

Required specifiers:

FACTORY=*object-name*
QBE.*op.attr_name*=*value*

Optional specifiers:

ALG_PRESET=*preset_expression*
ALG_PRESET_REL=*preset_expression*
CREATE_ALG=*activity_log_type*
HTMPL=*list_htmpl_file*
KEEP.*attr_name*=*value*

SEC_REFRESH

Refreshes the user access information from the security subsystem. A hyperlink for this operation is provided to users who have MODIFY privilege (for the admin functional group) on the menu screen. After updating a user's access privilege with the security program, this operation provides a means to refresh access information. (This refreshes the security information for all users.)

Note: Security refresh is an asynchronous process. When the security refresh is done, a message will be shown in the standard log file (stdlog).

SET_MENU

The behavior of this operation is the same as MENU when MENU is used with the HTMPL variable. The only difference is that this operation will also set the default menu form to the menu form specified with the HTMPL property.

Required specifiers:

HTMPL=*htmpl_file*

Note: This will override the MENU set in the web.cfg until the web service is restarted. See Configuration File Properties in the "Configure the Web Interface" chapter of the *Administrator Guide* for more information on web.cfg.

SHOW_DETAIL

Provides a generic interface to allow the user to display a read-only detail of a row in a specified table. The persistent ID name must be specified (from which the object name is inferred). By default, a template named detail_*xxx*_ro.htmpl is used for object *xxx*. The .htmpl file may be overridden by specifying the HTMPL property.

Required specifiers:

PERSID=*persistent-id*

Optional specifiers:

ALG_PRESET=*preset_expression*
ALG_PRESET_REL=*preset_expression*
CREATE_ALG=*activity_log_type*
HTMPL=*readonly_detail_htmpl_file*

UPDATE

Provides a generic interface to editing any table. The ID and object name must be passed in and an editable detail form is displayed to the user. By default, the user has exclusive access to the record for two minutes, and is guaranteed to get changes into the database if they are submitted in this time.

Required specifiers:

PERSID=*persistent-id* or
SET.id=*id-of-row-to-update* FACTORY=*object-name*

Optional specifiers:

NEXT_PERSID=*persistent-id* (of record to display after successful update)
KEEP.attr_name=*value*
KEY.attr_name=*value*
HTMPL=zdetailxxx_factory.htmpl

Note: In order to use the HTMPL specifier with UPDATE, the referenced form must have a name conforming to the naming convention shown above. It must begin with the string "zdetail", followed by any alphanumeric characters (including a null string), followed by an underscore and the factory name.

Operation Variables

This table lists the variables that can be set for each of the operations in the previous table:

Variables	Description	Operations
ALG_PRESET ALG_PRESET_REL	Specifies values for one or more of the attributes of the activity log created as a result of the CREATE_ALG variable. If CREATE_ALG is not specified, ALG_PRESET and ALG_PRESET_REL are ignored. See Syntax of ALG_PRESET and ALG_PRESET_REL following this table for a discussion of the syntax of this variable's argument.	CREATE_NEW SEARCH SHOW_DETAIL

Variables	Description	Operations
CREATE_ALG	<p>Specifies the activity log type of an activity log to be created as a side effect of the operation. Use the ALG_PRESET or ALG_PRESSET variables to specify values for the attributes of the new activity log.</p> <p>The timing of creation of the activity log depends on the operation, as follows:</p>	CREATE_NEW SEARCH SHOW_DETAIL
	CREATE_NEW	
	<p>The activity log is created when the new record is saved. If the new record is not saved, no activity log is created.</p>	
	SEARCH	
	<p>The activity log is created when a record is selected from the list form. If the record is viewed instead of selected (that is, the user explicitly selects the View command from the list form's mouse-over menu), no activity log is created.</p>	
	SHOW_DETAIL	
	<p>The activity log is created before the record is displayed.</p>	
FACTORY	<p>Specifies the class of object to be searched, created, or updated. You can use any name specified as an OBJECT in the *.maj files in \$NX_ROOT/bopcfg as listed in the appendix Objects and Attributes (see page 461).</p>	CREATE_NEW GENERIC_LIST SEARCH UPDATE
HTMPL	<p>Allows the HTMPL author to override the default template naming convention and explicitly specify the HTMPL file to display, instead of the default template.</p>	CREATE_NEW JUST_GRONK_I T MENU SEARCH SET_MENU SHOW_DETAIL
	<p>Note: When the HTMPL specifier is used with CREATE_NEW or UPDATE, the name of the referenced form must conform to the naming convention zdetailxxx_factory.htmpl, where xxx are any characters, and <i>factory</i> is the factory name.</p>	UPDATE
KEEP.name	<p>Specifies the value that can be saved and passed between pages.</p>	CREATE_NEW GENERIC_LIST SEARCH UPDATE

Variables	Description	Operations
KEY. <i>attr_name</i>	Similar to the SET. <i>attr_name</i> , except that this specifies a lookup on <i>attr_name</i> , which must be a reference to another table or object.	UPDATE
NEXT_PERSID	Specifies the persistent ID of the record to be displayed next.	UPDATE
PERSID	<p>Specifies the persistent ID of a record to be displayed. You can specify this in either of the following ways:</p> <p>Directly, with a persistent ID consisting of a factory name, a colon (:), and a unique integer database ID. For example, PERSID=chg:1234, specifies the change order with database ID 1234.</p> <p>Indirectly, with a persistent ID consisting of a factory name, a colon (:), an attribute name, a second colon (:), and a value. This form of PERSID specifies the record of the specified factory that has an attribute of the specified value. For example, PERSID=chg:chg_ref_num:demo:3 specifies the change order with reference number demo: 3.</p>	SHOW_DETAIL UPDATE
PRESET PRESET_REL	<p>Specifies values for one or more of the attributes of the record created as a result of the CREATE_NEW variable. If CREATE_NEW is not specified, PRESET is ignored.</p> <p>See Syntax of PRESET and PRESET_REL following this table for a discussion of the syntax of this variable's argument.</p>	CREATE_NEW

Variables	Description	Operations
QBE. <i>op.attr_name</i>	Specifies the values to use when performing a search. These values are identified using a QBE keyword, where <i>attr_name</i> identifies any attribute name on a ticket that can be set and <i>op</i> indicates to search where the attribute:	SEARCH
EQ	is equal to the value	
NE	is not equal to the value	
GT	is greater than the value	
LT	is less than the value	
GE value	is greater than or equal to the value	
LE	is less than or equal to the value	
NU	is null	
NN	is not null	
IN	matches the SQL LIKE expression	
KY	contains the text entered	
If you do not define any QBE variables, the standard search window is displayed.		
SET. <i>attr_name</i>	Specifies an attribute name to use when a ticket is created, where <i>attr_name</i> identifies any attribute in a ticket that can be set. The attribute names will vary depending on the underlying object. All objects and their attributes can be found in the *.maj files in the majic directory in bopcfg as listed in the appendix Objects and Attributes (see page 461).	CREATE_NEW UPDATE
SET. <i>id</i>	Specifies the database ID of the row to be updated.	UPDATE
SKIPLIST	When set to 1, searches that result in 1 hit do not display the search result list. Instead, the read-only detail is displayed directly.	SEARCH

Variables	Description	Operations
use_template	When set to 1, the SEARCH operation will return a list of templates. The returned template selected will be used in the CREATE_NEW operation to populate a new record. This variable is valid for change orders, issues, and requests.	CREATE_NEW SEARCH

Syntax of PRESET, PRESET_REL, ALG_PRESET, and ALG_PRESET_REL

The PRESET, PRESET_REL, ALG_PRESET and ALG_PRESET_REL keywords in the URL specify initial values for attributes of the ticket and its activity log, respectively. There are two possible formats:

[ALG_]PRESET=*attr:value*

Indicates that the specified attribute of the ticket or activity log should be set to the specified value. For example, the following specification sets the description of the new ticket to "Hello:"

```
PRESET=description>Hello
```

[ALG_]PRESET_REL=*attr:obj.relatr:testattr:value*

Indicates that the specified attribute of the ticket or activity log should be set to a value copied from another database table. The value is copied from the *relatr* attribute of the *obj* whose *testattr* has the specified *value*. For example, the following specification sets the analyst attribute of the new ticket to the ID of the contact with user ID xyz123:

```
PRESET_REL=analyst:cnt.id:userid:xyz123
```

When this format is used, the implied query must retrieve a unique record. If more than one contact has a user ID of xyz123 (or none), the example PRESET specification has no effect.

The PRESET, PRESET_REL, ALG_PRESET and ALG_PRESET_REL keywords can occur as many times as desired in a URL, allowing the setting of multiple attributes. Alternatively, a single keyword operand can specify multiple values separated by @@. If the '@@' separator is used, you cannot mix value formats for [ALG_]PRESET and [ALG_]PRESET_REL keywords. For example, the following example shows two different ways of specifying values for ticket description, summary and analyst:

```
PRESET=description>Hello+PRESET=summary>HelloThere+PRESET_REL=analyst:cnt.id:userid:xyz123  
PRESET=description>Hello@@summary>HelloThere+PRESET_REL=analyst:cnt.id:userid:xyz123
```

For requests, issues, incidents, problems, and change orders, both PRESET and PRESET_REL support a keyword attribute ASSET to link an object to an asset. The ASSET attribute updates the affected_resource attribute of a request, incident, or problem, or the asset LREL of an issue or change order.

Link Examples

The following example links omit the path to Unicenter Service Desk itself. All Unicenter Service Desk URLs begin with coding of the form:

`http://hostname[:port]/CAisd/pdmweb.exe`

where *hostname* is the name of your server, and *port* (optional) is the port number if you are using Tomcat. This coding is shown as an ellipsis (...) in the following URL examples:

- To create a new request with an affected end user with the userid tooda01
...?OP=CREATE_NEW+FACTORY=cr+PRESET_REL=customer:cnt.id:userid:tooda01
- To show a list of all requests assigned to userid tooda01
...?OP=SEARCH+FACTORY=cr+QBE.EQ.assignee.userid=tooda01
- To show the detail form for request 1234
...?OP=SHOW_DETAIL+FACTORY=cr+PERSID=cr:ref_num:1234 (read-only view)
...?OP=UPDATE+FACTORY=cr+PERSID=cr:ref_num:1234 (update view)

It is possible to bypass the logon challenge by using Web Services for authentication. See the `getBopsid()` method in the Web Services Guide.

Advanced Customization

The following sections describe various aspects of customizing web pages that you may need to be aware of if you elect to use tools other than Web Screen Painter to modify HTMPL, or if you have unusually complex customization requirements. However, we strongly recommend that you work with Web Screen Painter to customize Unicenter Service Desk web pages before trying any other approach. WSP is capable of doing almost any customization you need, and it automatically handles housekeeping issues, such as placing updates in the site mods directory, and distributing published files to all servers.

The Web Engine and Its Cache

When customizing web pages, it is helpful to understand the structure of the Unicenter Service Desk web server. The web interface uses either a J2EE servlet container, such as Tomcat, or a standard HTTP server, such as Apache or Microsoft Internet Information Server (IIS). When a user requests a Unicenter Service Desk web page, the HTTP server invokes the supplied program pdmweb.exe.

Once it starts, pdmweb.exe sets up a connection with a Unicenter Service Desk daemon (or Windows service) called the web engine. The web engine interprets the user's request. Most requests require the web engine to look up a template (HTMPL) file and translate it into standard HTML. Usually, the translation process requires the web engine to communicate with a Unicenter Service Desk server to read or update the database, and include database information in the generated HTML. Once the HTML is complete, the web engine sends it to pdmweb.exe, which in turn sends it back to the user's browser.

To maximize performance, the web engine normally reads each HTMPL file only once. After parsing the file and determining how to translate it to HTML, the web engine stores the parsed file in its cache, significantly reducing the processing time the next time the file is requested. While the cache is very beneficial in a production environment, it can be inconvenient in development, as it means that changes to HTMPL files do not take effect until either the web engine is recycled or the pdm_webcache utility is used. In a development environment, you can avoid this behavior by specifying the configuration file property SuppressHtmplCache (for more information, see in the chapter "Configure the Web Interface" of the *Administrator Guide*). However, suppressing the HTMPL cache is not recommended in a production environment because it severely impacts overall performance of the web engine.

The web pages served up by pdmweb.exe are generated by reading HTMPL files and using them to generate HTML. HTML template files are identified by a file suffix of .htmpl. You can modify these template files, and thereby customize the Unicenter Service Desk web pages.

The pdm_webcache Utility

Use the pdm_webcache utility to remove one or more HTMPL forms from the web engine cache. This forces the web engine to fetch these forms from the disk the next time they are used, allowing changes to forms to take effect.

```
pdm_webcache [-f form-name] [-g form-group] [-i interface] [-p process] [-v]
```

-f form-name

Specifies the name of the form to be removed from the cache, such as detail_cr.htmpl. You can use '%' (or '*') as a wildcard character to select more than one form. For example, the specification:

```
-f detail%
```

selects all detail forms.

This argument is optional. If it is omitted, all forms in the cache are selected.

-g form-group

Specifies the name of the form group to be removed from the cache, such as Analyst. You can use '%' (or '*') as a wildcard character to select more than one form group. For example, the specification:

```
-g Anal%
```

selects all form groups beginning with "Anal".

This argument is optional. If it is omitted, all form groups in the cache are selected.

-i interface

Specifies the name of the web interface to be removed from the cache, such as analyst, customer, or employee. You can use '%' (or '*') as a wildcard character. For example, the specification:

```
-i a%
```

selects the analyst interface.

This argument is optional. If it is omitted, all interfaces in the cache are selected.

-p process

Specifies the name of the web engine process whose cache is to be modified, such as web:local.

This argument is optional. If it is omitted, all web engines are selected.

-v

Specifies verbose output. When this argument is specified, pdm_webcache lists the full name of every form removed from the cache, in the form:

interface:form-group:form-name

This argument is optional. If it is omitted, pdm_webcache reports only a count of forms removed from each web engine's the cache.

How to Modify HTML Templates

Important! You must be familiar with HTML to modify the HTML templates.

Typically, you can make two types of changes to the HTML templates:

- You can make modifications that will be visible to the user but will not be altered by the web interface prior to display. For example, you could add a GIF file for your company logo to the web interface pages (a "pass through") by adding the reference to the appropriate template file or you could add JavaScript to your page to validate input. Any changes you make to the HTMPL file that are not contained within a PDM tag, as defined in the following, are passed unchanged in the HTML returned to the user.
- You can modify the replaceable sections of the templates. For example, you could add new application data to the request detail page.

Several kinds of template entries let you:

- Display information from Unicenter Service Desk to the user.
- Set up a query page.
- Create links to other Unicenter Service Desk pages using link tags.

Files That Should Not be Modified

Certain HTMPL templates and JavaScript files contain information required by many Unicenter Service Desk web forms. The information in these templates is both release dependent and critical to the successful operation of the Unicenter Service Desk web interface. Therefore, these files are always replaced when a new version of Unicenter Service Desk is released; changes made to them are not migrated.

The templates affected by this restriction are as follows:

ahdtop.htmpl

Contains styles, scripts, and JavaScript variables used throughout the Unicenter Service Desk web interface. This file is part of the main frameset of the web interface, and is always present during a session. All Unicenter Service Desk forms have access to the JavaScript variable ahdtop that references the window containing ahdtop.htmpl.

menu_frames.htmpl

Defines the HTML frameset used by the Unicenter Service Desk main form.

msg_cat.js

Contains the text of all messages used in Unicenter Service Desk JavaScript files.

reports.htmpl

Contains data required for web reports.

std_body.htmpl

Contains standard information used at the beginning of the BODY section of most HTMPL templates.

std_footer.htmpl

Contains standard information used at the end of the BODY section of most HTMPL templates.

std_head.htmpl

Contains standard information used at the beginning of the HEAD section of almost all HTMPL templates.

styles.htmpl

Contains CSS styles used throughout the Unicenter Service Desk web interface.

Although you cannot modify these files directly, you can add additional information to them. Each restricted file xxx.htmpl (except for menu_frames.htmpl and reports.htmpl) has a corresponding xxx_site.htmpl file that you can customize. For example, you can add additional information to ahktop.htmpl by customizing ahktop_site.htmpl, or add new messages by customizing msg_cat_site.js. The xxx_site.htmpl file corresponding to each restricted file is loaded after the main file so you can override or change JavaScript in the main file. Use caution when adding information, as badly designed changes to these files can cause unexpected problems throughout the Unicenter Service Desk web interface.

Guidelines for New HTMPL Files

You can add your own HTMPL files to the Unicenter Service Desk web interface. Follow these guidelines to help ensure your HTMPL file works well with the rest of the Unicenter Service Desk interface:

1. Include the following statement somewhere in the <HEAD> section of the file. This statement should follow the <TITLE> statement (if any). It defines several JavaScript global variables required by Unicenter Service Desk web interface, and also registers your page with the Unicenter Service Desk window manager:
`<PDM_INCLUDE FILE=std_head.htmpl>`
2. Include the following attribute as part of the <BODY> tag of the file. This attribute helps the Unicenter Service Desk window manager keep track of your page:
`onUnload="deregister_window()"`
3. Include the following statement at the beginning of the <BODY> section of your file. The “menubar=no” argument is optional; if specified, it suppresses the Unicenter Service Desk menu bar:
`<PDM_INCLUDE FILE=std_body.htmpl [menubar=no]>`
4. Include the following statement at the end of the <BODY> section of your file.
`<PDM_INCLUDE FILE=std_footer.htmpl>`

How to Add User Defined State Information

Many customers want to be able to embed their own state information in the Unicenter Service Desk web pages, and have Unicenter Service Desk pass the state information to all subsequent pages it serves up to the user's session. This information can be interrogated with conditional statements in the HTMPL files.

State information for a user's session is accomplished by setting the special attribute USER_STATE in your links or forms. Once submitted into the Unicenter Service Desk web engine, every page that is presented to the user will have the HTMPL variable USER_STATE available and set to the value last submitted for USER_STATE.

The following examples show how you might set up an entry into Unicenter Service Desk from some other part of your site, such as from pages that are oriented to your sales force:

- Using a hyperlink

```
<a href="/CAisd/pdmweb.exe?USER_STATE=Sales">Service Desk</a>
```

- Using a form with a hidden field

```
<form action="http://yourhost.com/CAisd/pdmweb.exe">
<input type=hidden name=USER_STATE value=Sales>
```

Click the button for the Service Desk

```
<input type=submit>
</form>
```

Then you can customize your HTMPL forms based on the state information:

```
<PDM_IF "$USER_STATE" == "Sales">
    custom information for sales audience
<PDM_ELIF "$USER_STATE" == "Engineering">
    custom information for engineers
<PDM_ELSE>
    information for everyone else
</PDM_IF>
```

How to directly create a Request from a Template

It is possible to create a new Request directly from a Template using an URL.

Example:

```
http://machinename/CAisd/pdmweb.exe?FACTORY=cr+OP=CREATE  
NEW+PERSID=cr:3106+use_template=1
```

where cr:3106 is the persid of the template.

Directories Used by Your HTTP Server

The default installation of Unicenter Service Desk defines two virtual directories to your HTTP server:

- The CAisd virtual directory points to the following directory in your Unicenter Service Desk installation:
 - In Windows: *installation-directory*\bopcfg\www\wwwroot
 - In UNIX: \$NX_ROOT/bopcfg/www/wwwroot
- The CAisd/sitemods virtual directory points to the following directory in your Unicenter Service Desk installation:
 - In Windows: *installation-directory*\site\mods\www\wwwroot
 - In UNIX: \$NX_ROOT/site/mods/www/wwwroot

Subdirectories under these virtual directories are:

Subdirectory	Stores
css	Style sheets
help	Web interface help
html	HTML files
img	Graphic files
scripts	JavaScript
sitemods	Site-defined customizations

If you decide to create a customized version any of the files in the css, html, img, or scripts directories, we strongly recommend that you do not update the file in /CAisd. Instead, store the file in the appropriate subdirectory of /CAisd/sitemods. For example, if you decide to modify a style sheet in /CAisd/css, store your customized version in /CAisd/sitemods/css. When the web engine parses an HTMPL file, it automatically modifies file names beginning with \$CAisd to point to sitemods if the file exists in a subdirectory of sitemods.

Using the /CAisd/sitemods directory has these advantages:

- It allows you to keep a record of the distributed files you have changed.
- It gives you easy access to the original version in case there is a question or a problem.
- It makes the process of installing maintenance or a new release easier, since Unicenter Service Desk installation never places anything in the /CAisd/sitemods directory.

Notes: There is no /CAisd/sitemods/help subdirectory. Because the help data is in standard HTML files (not HTMPL templates), the web engine cannot dynamically change file references. If you need to customize help, you must make your changes in /CAisd/help.

The HTML subdirectory contains a few heavily used files that do not need to be processed by the web engine and can improve performance when cached on the browser. If you create a customized version of any of these files, carefully check the file for references to other customized files. Because there is no web engine processing, you must manually insert a reference to sitemods where appropriate.

Looking Up Information in Reference Tables

Input fields on a detail form editing a database record are named `SET.attr_name`. When the record is saved, data from SET fields are copied directly to the underlying record. Thus, an input field for an attribute that references another table should contain the REL_ATTR (foreign key) of that table. This is normally the id, persistent_id, or code of the reference record.

Users do not directly provide REL_ATTR values, and the SET fields for attributes referencing another table are hidden. The visible field on the form is named `KEY.attr_name`, and it contains the common name of the referenced record. A common name must be converted into a REL_ATTR to update the record. There are several times when this might be done:

- For fields with a drop-down list, the SET value is provided directly by the drop-down.
- For fields with a lookup when the user clicks the lookup and selects an item, the SET value is copied from the selected item.
- For fields with a lookup where the user provides a partial key that uniquely identifies the record and then clicks the label, the browser requests the SET value from the server and copies both it and the full key back to the form.
- If the Autofill configuration file property is provided or defaulted, and the user both provides a partial key that uniquely identifies the record and clicks Notebook to exit the field, the browser requests the SET value from the server and copies both it and the full key back to the form.

Otherwise, when the record is saved with a KEY value and no SET value, the web engine resolves the value during the save. If any KEY values cannot be resolved to a unique SET value, the save is prevented, and the edit form is redisplayed.

If a form has been redisplayed as a result of a save that failed due to a lookup resolution failure, the following variables are available in the HTMPL for each attribute field for which a lookup was performed:

LIST_attr

Contains all the matches found. Typically this is specified as the right-hand side of the SOURCE= field in a <PDM_SELECT> statement.

FLAGS_attr

This is set to one of the following values:

0

Display initial search field.

1

More than one and fewer than MaxSelectList were found (typically a <PDM_SELECT> list would be displayed in this case).

2

No matches were found.

3

Too many matches were found (more than MaxSelectList).

SEARCH_STATUS_attr string

Contains the TooManyMatches text string from the web.cfg file (for more information on web.cfg, see Configuration File Properties in the “Configure the Web Interface” chapter of the *Administrator Guide*).

Specifying Lookups on Contacts

When specifying a contact (last name, first name, middle name) in an editable form, you can delimit the contact name with commas (,) or blank spaces, but not both. Commas are preferable because names often have embedded spaces, which cause problems.

Since a combination of commas and blank spaces is not allowed, the presence of commas implies that all parts of the name are comma-separated; if no commas are present, names are delimited by spaces.

Since the information is eventually passed to an SQL query, the percent symbol (%) serves as a wildcard character. For example, 'P%, J%' would match 'Public, John', 'Penxa, Jane', and any other names whose last name begins with P and first name begins with J. (Case-sensitivity depends on the underlying database.) Similarly, 'P% J%' would bring up the same names.

However, 'P%, Jon D' would not bring up all contacts with a first name of Jon, a middle initial of D, and a last name beginning with P, because the presence of one comma means all delimiters are commas. Therefore, the last name would be looked up as 'P%' and the first name would be looked up as 'Jon D'. To avoid this error, specify 'P%, Jon, D' instead.

Web Engine PreProcessing

As discussed in The Web Engine and Its Cache above, the web engine goes through two phases when processing an HTMPL file:

- The preprocessing phase, when it reads the HTMPL file and any referenced files (including files referenced by PDM_INCLUDE and PDM_MACRO tags). The output from preprocessing is an entry in the web engine's internal cache.
- The generation phase, where it reads the form from its cache and generates HTML. The output from generation is HTML delivered to the browser.

The pre-processing phase is typically done once for each form in the lifetime of the web engine. The generation phase is done each time a form is requested.

You can use the PDM_SET and PDM_EVAL tags during the preprocessor phase to generate and store information, such as HTML text, that the web engine can use in the generation phase.

Preprocessor Variables

Preprocessor variables begin with the string "\$PRE.". They are created and updated with the PDM_SET (see page 154) tag. This tag has the following syntax when used with a preprocessor variable:

```
<PDM_SET PRE.name[+]=value>
```

This tag assigns or updates a preprocessor variable, creating it if necessary. It is processed when the web engine encounters it while reading a form. Only the invariant PDM_IF statements described in the next section affect PDM_SET of a preprocessor variable; others are ignored.

Invariant PDM_IF Detection

When parsing a form, the web engine detects invariant PDM_IF statements. An invariant PDM_IF is one whose argument consists entirely of literals, environment variables, constant properties, and preprocessor variables. When the web engine detects an invariant PDM_IF, it evaluates its condition immediately. This has the following effects:

- PDM_SET and PDM_EVAL tags that are bypassed by an invariant PDM_IF are ignored. Note that all other pdm_eval tags and PDM_SET tags referencing preprocessor variables are executed when processed, even if they are within a non-invariant PDM_IF.
- Form variable references bypassed by an invariant PDM_IF are ignored, and their value is not fetched when the form is used. You can use this technique to improve the performance of a form. For example, if a form contains the following, the web engine fetches the value of \$args.def before it displays the form:

```
<PDM_IF "$env.NX_OTB_MARKET == "itil" && "$args.a" == 1>
<h1>This is form $args.def</h1>
</PDM_IF>
```

However, if the following segment has been written, the web engine determines that the first PDM_IF is invariant, and retrieves the value of \$args.def only if \$NX_OTB_MARKET is "itil".

```
<PDM_IF "$env.NX_OTB_MARKET == "itil">
<PDM_IF "$args.a" == 1>
<h1>This is form $args.def</h1>
</PDM_IF>
</PDM_IF>
```

PDM_EVAL: Insert Text from a Preprocessor Variable

The PDM_EVAL tag inserts the value of a preprocessor variable into the input to the web engine parser. If used inside a macro, its effect is deferred until the macro completes.

The PDM_EVAL tag works similarly to PDM_INCLUDE or PDM_MACRO. It inserts the text into the parser at the point of the tag, exactly as if the value of its variable had been coded in place of the tag.

PDM_EVAL has the following syntax:

```
<PDM_EVAL text=PRE.name>
```

where PRE.name specifies the name of the preprocessor variable whose value is to be inserted into the web engine's input

Execution of the PDM_EVAL tag can be controlled by invariant PDM_IF statements.

Free-Form Customization of Detail Forms

Using JavaScript on Detail Forms

You can use Web Screen Painter to add your own fields to a detail form, or rearrange or change edit characteristics of fields provided on the form by default. However, sometimes you want to customize a form beyond simply adding new fields to a grid. There are a number of JavaScript functions provided with Unicenter Service Desk to make it easy to merge your own customizations into a combination detail form and give it any appearance you want. These functions are summarized below, and are documented in detail in the next section.

- You may freely place any HTML whatsoever prior to the `DetailForm()` statement or after the `endDetail()` statement without affecting the operation of the detail form at all.
- You can use the `detailEndTable()` function to close the table that lays out detail form elements in a grid. Once you have done this, you can lay out your own HTML in any desired format. In this case, your HTML is inside the detail form, and any form fields within it are submitted to the web engine when the user clicks Save. You can use the `detailNextID()` function to generate ID fields for your HTML elements that allow them to participate in mouse-less navigation of the detail form. You can see several examples of this technique in the notebook tabs, such as `xx_alg_tab.htmpl`.
- You can follow your own HTML with a `dtlStartRow` macro to restart standard detail form formatting. This starts a second grid, whose fields will not necessarily be aligned with the first. This technique is used in every notebook tab.
- If you want to insert a custom element at the end of a row, you can use the `detailWriteRow()` function to write out the contents of a row without closing it. You can see an example of this technique in the code that generates the "24 Hour" button in `detail_cr.htmpl` and `detail_iss.htmpl`.
- If you want to explicitly specify the contents of an element in a row without closing out the table that lays out the grid, you can use the `detailRowHdr()` function to specify the header text and the `detailSetRowData()` function to specify the data text. You can see an example of this technique in the code that generates the timer field in `detail_cr.htmpl` and `detail_iss.htmpl`.
- If you provide a function to validate a field's value (normally in an event handler), and want its results reported during browser-side validation (so that an erroneous field is redrawn with a thick red border, and an error message appears in a yellow band at the top of the form), use the function `detailReportValidation()`. You can see an example of this in the `validate_duration()` function used to validate the duration fields in `xx_candp_tab.htmpl`. The `validate_duration()` function is in the file `val_type.js`.

- If you want review the HTML generated for a detail form, you can use functions docWrite() and docWriteIn() in place of the standard functions document.write() and document.writeln(). Then if you invoke the function holdHTMLText() anywhere in the <HEAD> section of your form, Unicenter Service Desk will pop up a debugging form containing a TEXTAREA with all of the HTML generated for the form, which you can review or copy and paste into a validation tool.

While you are composing your modifications, remember that the combination detail form is displayed in both a read-only and an edit view. If your customizations apply specifically to one view or the other, you can test the current view in one of two ways:

- In JavaScript, the expression _dtl.edit is true in the edit view and false in the read-only view.
- In either JavaScript or open HTML, the statements:

```
<PDM_IF "$prop.form_name_3" == "edit">  
  (code used only in the edit view)  
</PDM_IF>  
  
or  
  
<PDM_IF "$prop.form_name_3" == "ro">  
  (code used only in the read-only view)  
</PDM_IF>
```

can be used to bracket code intended only for the edit or read-only view, respectively.

detailEndTable()

This function closes the HTML table that lays out the detail form elements in a grid. It has no arguments.

You can start a new grid with the dtlStartRow() macro. However, elements in a new grid are not necessarily aligned with elements in a previous grid.

detailNextID([colspan,] [lastelement])

This function returns a string of the form:

```
" ID=df_nn_nn TABINDEX=n onFocus=func onBlur=func"
```

Inserting this string into an HTML element causes the element to follow the conventions of Unicenter Service Desk mouse-less navigation, including accessibility with the arrow keys and turning pale yellow when focused. Note that the returned string begins with a space and ends with no space.

colspan

Specifies the number of columns in the grid occupied by the element. This argument is optional; it defaults to one if not provided. If omitted, the element is assumed to occupy one column of the grid. This affects arrow key behavior. The *colspan* argument can be omitted even if the *lastelement* argument is provided.

lastelement

A Boolean value specifying whether the element for which the ID being generated is the last one in its row. If omitted, the element is assumed to be followed by other elements. This affects arrow key behavior.

detailNextLinkID()

This function returns a string of the form:

```
" ID=dflnk_nn_nn TABINDEX=0 onFocus=func onBlur=func"
```

Inserting this string into an HTML element defining a link element causes the element to follow the conventions of Unicenter Service Desk mouse-less navigation, including accessibility with the up arrow key from the base element and turning pale yellow when focused. Note that the returned string begins with a space and ends with no space.

This function takes no arguments.

detailReportValidation(field, has_error, emsg)

This function reports the result of external field validation. If validation is reported to have failed, the field is redrawn with a thick red border and the error message provided is shown in a yellow band at the top of the form. The user is not permitted to save the record until a subsequent call to detailReportValidation() reports the field as error-free.

The detailReportValidation() function is functional only for fields registered for browser-side validation. All fields created with detail form macros are automatically registered for validation. You can register other fields with the detailSetValidateFunction().

field

Specifies the form element object containing the field. The easiest way to obtain this is to pass this argument to the event handler performing the validation. Another way is to use the standard JavaScript function document.getElementById(). This argument is required.

has_error

A Boolean or integer value specifying whether the field is in error. Setting a field in error prevents the user from saving the record, causes the field to be highlighted with a thick red border, and places the error message supplied as the third argument in a yellow band at the top of the form. Setting a field as not in error reverses these changes. This argument is required.

emsg

A text string specifying the message to display in the yellow band at the top of the detail form when the *has_error* flag is set. This argument is required if *has_error* is set.

detailSetValidate(hdrtext, is_required, maxsize)

This function specifies that the most recent field created with an ID supplied by detailNextID() is subject to browser-side validation. Validation for required fields and for fields with a maximum size is automatic. Other forms of validation may be provided through JavaScript functions or event handlers calling detailReportValidation().

You should call detailSetValidate() only for form fields you have defined yourself whose ID was created by detailNextID(). The detailSetValidate() function must be called immediately after creating a field that you want validated. It is unnecessary (and will cause unexpected results) to call detailSetValidate() for fields created by detail form macros.

hdrtext

Specifies a string used to identify the field in error messages. This argument is required.

is_required

A Boolean or integer value specifying whether the field is required. Unicenter Service Desk automatically verifies that all required fields are provided whenever the user attempts to save a record. This argument is required.

maxsize

An integer specifying the maximum length of data allowed for the field. Unicenter Service Desk automatically verifies that all fields with a *maxsize* value have a length within limits whenever the user attempts to save a record. This argument is required. To suppress *maxsize* validation, specify a value of 0.

detailRowHdr(*hdrtext*, *colspan*, *is_required*)

This function stores text for the header (TH) element of an item in the grid. The text is not actually written to the form until a detailWriteRow() function or dtlStartRow macro is invoked.

hdrtext

Specifies the text in the header element. This argument is required.

colspan

Specifies the number of columns in the grid occupied by the element. This argument is optional; it defaults to one if not provided. If omitted, the element is assumed to occupy one column of the grid. This affects arrow key behavior. The *colspan* argument must be provided if the *is_required* argument is provided.

is_required

Specifies whether the *hdrtext* should be displayed in the style corresponding to a required field. The argument can be a Boolean, a number, or a string. A number or a string is interpreted as false if zero and true otherwise. This argument is optional; if omitted, the *hdrtext* is styled as a non-required field.

detailSetRowData(*text*)

This function stores HTML text for the data (TD) element of an item in the grid. The text is not actually written to the form until a detailWriteRow() function or dtlStartRow macro is invoked. The single argument is the HTML text of the element to be stored.

detailWriteRow()

This function writes the HTML stored for the current row. This creates two HTML table rows, one for the header (TH) elements and one for the data (TD) elements. The function also writes the <TD> tag that begins a new data element. The TD tag is automatically closed by the dtlStartRow macro, so it is unnecessary (and incorrect) to provide the <TD> tags in HTML text that follows detailWriteRow(). This function has no arguments.

Understanding List Forms

Overview

Note: The following provides background information on the internals of USD list forms. We recommend that you use WSP Design View as specified in Edit List and Detail Forms in Design View (see page 106) to modify these forms.

Unicenter Service Desk list forms are defined with the following macros (invoked with the PDM_MACRO tag):

lsStart

Begins a list

lsCol

Defines a column in a list

lsWrite

Inserts text into the pdm_list part of a list (see below)

lsEnd

Ends a list

The general form of a list using these macros is:

```
<pdm_macro name=lsStart>
<pdm_macro name=lsCol hdr=hdr1 attr=attr1>
<pdm_macro name=lsCol hdr=hdr1 attr=attr1>
<pdm_macro name=lsEnd>
```

which results in text similar to the following in the output HTML:

```
var rs = new Resultset();           From lsStart
rs.startList(); From lsStart
rs.header("hdr1"); From lsCol
rs.setData("attr1","options"); From lsCol
rs.header("hdr2"); From lsCol
rs.setData("attr2","options"); From lsCol
<PDM_LIST SOURCE=list> From lsEnd
rs.data(attr1) From lsCol/lsEnd
rs.data(attr2) From lsCol/lsEnd
</PDM_LIST> From lsEnd
```

Note: There are two distinct sections to the output list: the setup section before the <PDM_LIST> tag, and the actual list between the <PDM_LIST> and </PDM_LIST> tags. The lsCol macro makes use of preprocessor variables and the <PDM_SET> tag to output data to both sections of the list. The entire list section of the list is created by a <PDM_EVAL> tag generated by the lsEnd macro.

To insert your own JavaScript in the setup section of the list, simply include it where needed. Use the IsWrite macro documented below to insert your own code into the list section of the list.

The IsWrite Macro

The IsWrite macro specifies text for the list section of a list (the portion between the <pdm_list> and the </pdm_list> tags). Text specified for the text argument of this macro is deferred, and not written to the output HTML until the IsEnd macro.

IsWrite [both=no|yes]

text="xxx"

both

Specifies that the text operand is to be written both immediately to the output HTML and to the deferred text buffer. This can be useful to output JavaScript to conditionally bypass both the setup and the list information output by a subsequent IsCol macro. Optional; defaults to no.

text

Specifies the text generated by this macro. Text specified is deferred until the IsEnd macro.

It is often desirable to include pdm tags and references to form variables in the text output by an IsWrite macro. To prevent these from being interpreted by the web engine during parsing of the IsWrite macro itself, follow these syntax rules:

- If the IsWrite macro generates a pdm_tag, omit the surrounding "<" and ">" delimiters of the tag. For example, to insert a <pdm_else> statement into the list section of the list, code:

```
<PDM_MACRO NAME=lsWrite text="pdm_else">
```

The web engine automatically inserts the "<" and ">" before producing the text when it detects that the first four characters are "pdm_" (or "PDM_").

- If the IsWrite macro generates a reference to a form variable, code an @ character in place of the \$ character that designates the variable. For example, to generate a reference to the list variable \$list.persistent_id, code:

```
<PDM_MACRO NAME=lsWrite text="@list.persistent_id">
```

The web engine automatically converts the "@" to "\$" before producing the text. To produce a literal @ sign, precede it with a backslash.

Customizing the Edit in List Feature

Several list forms, such as the Request and Issue lists, include an Edit in List button. When this button is available and a result set is displayed, the user can click Edit in List to replace the search filter with a small edit form. The edit form allows the user to update records directly on the list form. The user can even update everything selected in the list by placing the desired new data in the edit form and clicking Change All.

Editing list data involves no communication with the server until the user clicks Save. When the user clicks Save, all of the updates (marked by yellow highlights on the form) are sent to the server, which applies all the changes in a single operation, returning a status message and redisplaying the list.

You can customize this feature by controlling whether the Edit in List button is available on a particular list form, and by controlling the fields that appear in the edit form displayed when the user clicks Edit in List.

To place an Edit in List button on a list form, including the following statement somewhere in the <HEAD> section of the form:

```
<SCRIPT LANGUAGE="JavaScript" SRC=$CAisd/CAisd/list_edit.js></SCRIPT>
```

Simply adding this statement puts the button on the form. However, the button is disabled unless JavaScript statements specifying the contents of the edit form are also included in the form. These statements must be placed immediately prior to the results set specification, and have the following format:

Statements	Comments
startListEdit(_search_filter) ;	Specify exactly as shown
listEditStartRow();	Specify exactly as shown
listEditField("attr"[, "hdr"]);	Specify zero or more
listEdit_READONLY("attr[", "hdr"]);	Specify zero or more
endListEdit();	Specify exactly as shown

The endListEdit() statement must be followed by the ResultSet() statement that begins the results set. You specify the fields in the edit form and their sequence on the form by coding one or more listEdit_READONLY() or listEditField() statements.

The list edit definition statements you can customize are as follows:

`startListEdit(_search_filter);`

This statement begins the list edit form. It must be coded exactly as shown.

`listEditStartRow();`

This statement begins a new row of fields on the list edit form. It must be coded exactly as shown. You must place a `listEditStartRow()` statement immediately after the `startListEdit()` statement. You can optionally include additional `listEditStartRow()` statements among the `listEditField()` and `listEditReadonly()` statements that specify the fields on the form.

`listEditField("attr_name" [, "hdr"]);`

This statement specifies an attribute to be included on the list edit form.

attr_name

Specifies the name of the attribute to be included in the edit form (including dots, if appropriate). All attributes specified for a list edit form must also be in the results set. The `attr_name` specified must be identical to that specified in the `rs.showData()` or `rs.showDataWithLink()` that adds the attribute to the results set.

The attribute appears on the edit form in the same format that it appears in the search filter. If the attribute is not in the search filter, it is edited in a 20-character text box.

`attr_name` is a required argument.

hdr

Specifies the text of the header on the field in the edit form. This argument is optional; if omitted, the header text is taken from the search filter. If `hdr` is omitted and the attribute is not in the search filter entry for `attr_name`, the header text defaults to the attribute name surrounded by question marks.

`listEditReadonly("attr_name" [, "hdr"]);`

This statement specifies a non-editable attribute to be included on the list edit form. Its arguments have the same significance as those for `listEditField()`.

`endListEdit();`

This statement ends the list edit form. It must be coded exactly as shown.

Integrating with Your Own Web Pages

This section describes how you can integrate the Unicenter Service Desk web interface functionality with your web pages to present a seamless interface for your users.

Note: The web engine, which is the executable that acts as the gateway between the web server and the Unicenter Service Desk server, allows multiple simultaneous connections from a given user. More than one frame at a time can have an open connection to the Unicenter Service Desk web engine process.

You can integrate the web interfaces in the following ways:

- By creating links from any of your web pages to the appropriate Unicenter Service Desk web page without having to go through the web interface menu page.
- By adding HTML forms to your web pages that collect input and perform supported operations (see page 163) directly, without displaying any Unicenter Service Desk web data entry pages.
- By creating web form groups that can be used to associate HTML web-based forms to users through their access type. Similar to the form groups used by the administrative client, web form groups can be used to customize your HTML pages.

Linking to Unicenter Service Desk Functions

You can link directly to major Unicenter Service Desk functions without displaying the main page. You normally do this by accessing the screen pop for the new window containing the Unicenter Service Desk information. Or, you can also replace your web page with the Unicenter Service Desk page.

In both cases, Unicenter Service Desk displays the requested page in the same way that the user sees it in a normal session but without the main page and scoreboard. If you are an analyst, display the main page and scoreboard by selecting Restore Scoreboard from the File menu, which is available only on pages displayed by bypassing the main page.

To create a link that bypasses the main page, specify a URL of this form:

`http://hostname[:port]/CAisd/pdmweb.exe?OP=operation+var=value+...`

where *hostname* is the web server host machine; *port* is the port number (usually 8080) required only if you are using Tomcat as your http server; *operation* is one of the supported operations (see page 163); and *var=value* is one or more of the variables allowed with the operation).

For example, a link that loads the form for creating a new request can be specified as:

```
<A HREF="http://hostname/CAisd/pdmweb.exe?OP=CREATE_NEW+FACTORY=cr>Define  
Request</A>
```

The Link Examples (see page 172) topic lists additional examples.

Posting Forms to Unicenter Service Desk

You can also access Unicenter Service Desk functionality by adding HTML forms to your web pages that refer to supported operations (see page 163). If the form is submitted with sufficient information to perform the operation, such as creating a request, the operation is performed without displaying a form to collect additional input.

When you add an HTML form to your web page:

- The ACTION for the form is the URL for pdmweb.exe.
- The METHOD is POST.
- Either the name of the SUBMIT button should be one of the supported operations, or you should have a hidden field named OP whose value is one of the supported operations (see page 163).

For example, to create an HTML form that loads the page for creating a new request, specify the following code:

```
<FORM ACTION=/CAisd/CAisd/pdmweb.exe METHOD=POST>
<INPUT type=HIDDEN NAME=FACTORY VALUE=iss>
.
.
.
<INPUT type=SUBMIT NAME=CREATE_NEW VALUE=" OK ">
</FORM>
```

Customizing JavaScript

The Unicenter Service Desk web interface makes extensive use of JavaScript and includes a number of JavaScript files in the /CAisd/scripts directory. If you decide to customize any of these script files, place the modified version in /CAisd/sitemods/scripts, as already described in Directories Used by Your HTTP Server (see page 179).

For performance reasons, the JavaScript files delivered in the /CAisd/scripts directory are compressed, with comments and unnecessary white space removed. This can make them difficult to read. You can find uncompressed versions of all JavaScript files in one of the following directories:

\$NX_ROOT/sdk/scripts (UNIX)

or

\$installation-directory\ sdk\ scripts (Windows).

If possible, though, you should avoid creating customized versions of entire JavaScript files, since each file contains a number of functions and you may only wish to modify one. In most cases you can override individual functions by placing a modified version in the JavaScript file sitemods.js. We strongly recommend you take this approach to modifying JavaScript.

How to Use sitemods.js

A skeleton sitemods.js file is distributed with Unicenter Service Desk. All distributed HTMPL files include this file at the end of their <head> section, making it the last JavaScript file loaded. Because it is the last file, any functions defined in it override functions with the same name included earlier. This lets you provide your own version of a distributed JavaScript function without directly modifying distributed code.

This approach is not effective for functions invoked at load time in the <head> section, such as those in menubar.js and ahdmenus.js. (For more information about editing the Menu Bar, see Edit Menu Bars). However, most JavaScript functions can be customized with the following steps:

1. Place a modified version of the function in sitemods.js.
2. Store the updated copy of sitemods.js in
CAisd/site/mods/www/wwwroot/scripts.

Functions Useful in Menu Items

Unicenter Service Desk provides a menu bar on almost every form to control its functions. The menu bar is generated by an HTMPL form with a name of the form menubar_xx.htmpl. We recommend that you use Web Screen Painter to customize existing menu bars and define new ones.

The following predefined functions may be useful for scripts invoked by menu items:

upd_frame(form)

Loads a new form into the main window content frame.

create_new(factory, use_template, width, height [,args])

Pops up a form to define a new record.

Popup_window(name, form[, width, height [,features [,args]]])

Pops up a new window.

showDetailWithPersid(persid)

Pops up a detail record.

The following terms and definitions apply to the previous functions:

form

This is either an HTMPL file name of the form xxx.htmpl or an operation code (for example CREATE_NEW).

factory

This is the name of a database object.

use_template

This is either true or false.

width

This represents the desired form width or zero for default.

height

This represents the desired form height or zero for default.

features

This is a list of window features, in the same format used with the standard window.open function.

args

This is one or more args of the form "keyword=value" for the operation specified for form.

persid

This is a persistent ID in the form factory: ID.

Modifying Context Menus

A number of forms within Unicenter Service Desk use context menus, accessed by right-clicking on an object. It is possible to modify context menus to add, remove, or modify their items. This can be done with the following two functions in sitemods.js

siteContextMenuItemAdd()

This function is called once after building a context menu. The syntax appears as follows:

```
function siteContextMenuItemAdd(menuName,menu)  
{  
}  
}
```

Use this function to add new items to the end of the menu. To add one or more items, issue one or more calls of the following form:

```
menu.addItem( text, script );
```

The following terms and definitions apply to the previous function:

Text

This represents the text you want to place on the menu.

Script

This is a string containing JavaScript to execute when the menu item is selected.

Note: The siteContextMenuItemMod() function is called for each menu item you add.

Return Value

There is no return value.

siteContextMenuItemMod()

Use this function to rename or eliminate the entry. It is called once for each entry in a context menu, as described by the following syntax:

```
function siteContextMenuItemMod(menuName, itemName)

{
    return true;
}
```

Some of the context menu names used in Unicenter Service Desk are as follows:

idGraphFolder

This is the scoreboard folder menu.

idGraphItem

This is the scoreboard item menu.

idGraphAll

This is the scoreboard combined menu.

idDetailsMenu

This is the menu on a search results set.

rClickMenu

This is the right-click menu on all forms.

The following Return values apply to this function:

True

Add the menu item.

False

Do not add the menu item.

"xxxx"

Rename the menu item to xxxx.

Updating and Creating Change Orders as Employee User

By default, a user can only view change orders from the Employee web interface. Use the following steps to enable creating and updating change orders by employees:

1. Sign on to the web as the Administrator, and select the Administration tab.
2. Select Access Type from the Security menu.
The Access Type List appears.
3. Select the Employee link to display the Employee Access Type Detail window.
4. Set the Change Orders to "modify" under the Function Access tab and save.
5. Click the Back button to return to the Administration tab, and then select Data Partitions, Data Partition List.
6. Click Employee to display the Data Partition Detail window. On the Constraints List portion of the window, review the Type column for following Change_Request Tables:
 - Pre-Update
 - Create
7. For each Table that you want to edit, click the Table name to display that table's Data Partition Constraint Detail window.
8. Click the Edit button.
9. Edit the constraint as follows:
`change "id = 1" to "affected_contact = @root.id".`
10. Click Save.

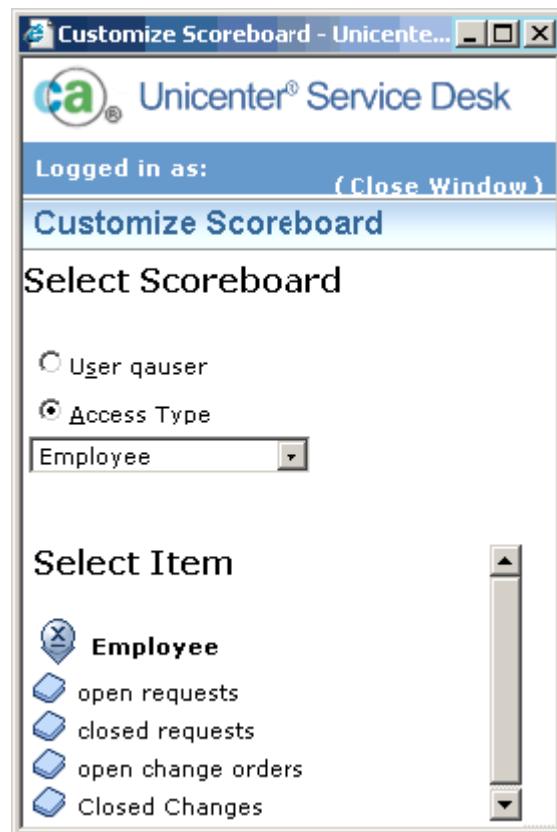
Now, when you login to the web interface as an employee user, the *Create Change Order* link will appear.

Add "Closed Change Orders" link to the Employee Scoreboard

To add a *Closed Changes* node option to the Employee web interface scoreboard:

1. Login as Administrator, and on the Service Desk tab select Customize Scoreboard from the File menu.
2. Select the Access Type radio button, and from the drop down list below it select Employee.
3. Click the Nodes Stored Query link, and search and select "Closed Changes" from the Stored Query List. Typically, Code: CHGUBIN7.
4. To indicate a location for the new node, select an item in the scoreboard tree on the left. The new item will be added under the item selected.
5. Click the Add New Node button.

The new node, "Closed Changes", is added to the Employee user scoreboard tree.



Download Attachments

When you download an attachment in Service Desk, it automatically displays the attachment in the browser window without prompting for a response from you. This could be dangerous in the event a virus is associated with the attachment.

With Service Desk you have the option to force a save-as dialog prompting you to respond if you want to save the attachment on the disk or open it. This can be a secure method as you can save the attachment on the disk and scan it before you can actually open it. You also have the option to force the save-as dialog only on certain attachment types.

This can be done through the web.xml servlet configuration file. The web.xml file can be found at the following paths:

Windows: NX_Root\bopcfg\www\CATALINA_BASE\webapps\CAisd\WEB-INF\web.xml

Linux: NX_ROOT is "/opt/CAisd"

Chapter 7: The Screen Painter

You can customize forms defined in the database to meet the needs of your enterprise using Screen Painter. Screen Painter is an interactive editing tool that lets you add, delete, move, or change items on a form, such as menus, fields, buttons, grids, notebooks, and so forth. Screen Painter is used to modify forms only for the Service Desk Java Client. To modify the Web client interface, use Web Screen Painter, described elsewhere in this guide.

Most tasks and examples in this chapter focus on customizing existing forms. You can also create entirely new forms using Screen Painter.

Note: It is easier and faster to copy and then edit an existing form that most closely resembles the form you want than to create a new form.

Important! Controls (see page 212) placed in a form in Screen Painter do not provide a functionality if the functionality is removed by customized back-end logic (majic, spell, or wand). Instead, it denies access to the functionality only. For more information about controls placed in a form in Screen Painter, see the Screen Painter online help.

The Issue Management Function

The Issue Management function of Unicenter Service Desk is a graphically based business process application, developed using an object-oriented distributed client/server environment. You can customize this application, including the database schema, object schema, and forms defined in Screen Painter, to best suit your needs.

Configuration Overview

The architecture of Unicenter Service Desk consists of the following tiers:

- Database schema
- Object schema (definition and behavior of objects)
- Application level viewer to manipulate objects (graphical user interface)

The first two tiers represent the server layer, and the last tier represents the client layer.

Database Schema

The database schema, or *backstore*, defines database tables, the fields (columns) in a record (row) of each table, the relationship among these records, and how they are sorted and indexed in the database. This backstore provides persistent (that is, continuous and lasting) storage and access for the data contained in *objects*.

Object Schema

Objects represent specific business knowledge and functionality. Each object contains data (attributes) and behavior (methods and triggers) and each associates with other objects by containment or other types of relationships. Objects generally correspond to database schema specifications. For example, an object attribute typically corresponds to a field in a database table.

Objects are assembled into and manipulated by a domain server. The domain server (domsrvr) uses the backstore as a repository for persistent objects. Object configuration files (Majic language files with a .maj extension) are interpreted at domsrvr startup time.

Objects are distributed. The domain server executes on one computer on a network (server), and one or more *viewers* execute on other computers (clients) on the network. Domain server objects and other special services exist concurrently and communicate using an underlying asynchronous messaging protocol that is transparent to most programmers.

Application Level Viewer

The object viewer displays domain server objects in a graphical manner. The viewer is the Unicenter Service Desk java client, in which all of the windows are defined as Screen Painter forms.

Database Schema Customization

You can modify the flexible database schema to meet your needs. For example, you can add a field to a table or add a new table.

Specific database tables, fields, and attributes are defined in schema (.sch) files. All schema files are stored in the \$NX_ROOT\site directory (UNIX) or *installation-directory\site* directory (Windows). Unicenter Service Desk provides an additional subdirectory within the main site directory called mods in which to store product modifications, including any changes or additions you make to the database schema. You should copy all of your changes into this mods subdirectory—it is never overwritten when you install a new version of Unicenter Service Desk.

When you configure Unicenter Service Desk, all of the schema files, including the new ones you create and copy into mods directory, are merged into ddict.sch. This contains the definition of the provider, partition, tables, storage details, and mappings for the backstore.

Object Schema Customization

You can also modify the object schema to meet your needs. For example, you can customize existing objects or add new objects. *Majic* is the metalanguage that defines the objects; therefore, objects are defined in Majic (.maj) files. The appendix “Object Definition Syntax” (see page 705) describes Majic language syntax.

All Unicenter Service Desk object definition files are stored in the \$NX_ROOT/bopcfg/majic directory (UNIX) or *installation-directory\bopcfg\majic* directory (Windows). Unicenter Service Desk provides an additional directory called \$NX_ROOT/site/mods/majic (UNIX) or *installation-directory\site\mods\majic* (Windows) in which to store any changes or additions you make to the object schema. You should copy all of your object schema changes into this mods subdirectory—it is never overwritten when you install a new version of Unicenter Service Desk.

These object configuration files are interpreted at domsrvr startup time. When the domsrvr starts, it reads the object definition files in the \$NX_ROOT/bopcfg/majic directory (UNIX) or *installation-directory\bopcfg\majic* directory (Windows) first, and then searches the \$NX_ROOT/site/mods/majic (UNIX) or *installation-directory\site\mods\majic* (Windows) directory for modifications.

The appendix “Objects and Attributes” (see page 461) lists all the objects and attributes that define Unicenter Service Desk.

Forms Customization

You can edit most of the form definitions contained in the database to customize the java client using Screen Painter. You can also create a new form to display data from an existing object or to display data from a new database table and object you created. The sections that follow give you a brief introduction to Screen Painter, and the remainder of this chapter goes into greater detail about how to use Screen Painter.

Tags

Screen Painter uses *tags* to identify:

- The name of an object
- An attribute to display or enter
- A method to run

A tag consists of an object name and an attribute name, in the format *object.att_name*. One example is *iss.open_date*, which provides the open date in the Date field on the Issue Detail window. Another example is *agt.contact_num*, where *contact_num* is an attribute in the *cnt* object and *agt* is a *factory* of the *cnt* object. A factory is a subset of an object (that is, a different view of the object). For example, the *cnt* (contact) object has three factories: *cst* (customer), *agt* (analyst), and *grp* (group).

The tag can include additional attributes in the format *object.att1_name.att2_name.att_name*. The additional attributes are part of related objects. This relationship is based on the SREL (single relationship) parameter, which refers an attribute to another object. An example of this is *cnt.location.id*, where *location* is an attribute in the *cnt* object that is defined with SREL *loc*, which means it refers to the *loc* object, and *id* is an attribute in the *loc* object.

The selection of a tag is very important. If the tag name is entered incorrectly, the object, attribute, or method cannot be accessed and the field is unusable. Screen Painter contains a Tag Helper that displays the objects, attributes, and methods that are currently available. It also shows the relationships among objects and builds the correct tag name for you. You can select a tag name from the Tag Helper tree and apply it to a field on your form.

Note: You are not allowed to manually type in tag content that contains spaces. Usually, a tag shouldn't contain any spaces. However, in very rare situations where you need to add descriptions to a tag for report purposes, you do need spaces in the tag. In such situations you can copy and paste a string with spaces to the tag field of a control.

Important! In Screen Painter, the only way to refresh the tag helper to pick up new objects when the object schema has been modified is to delete the \$NX_ROOT\site\dispobs file before invoking Screen Painter.

Relationship between Form Groups

Form groups are used to control which users are allowed to view which forms. The administrator for your service desk creates form groups, assigns them to access types, and assigns access types to users in the java client, and when a user accesses the java client, their form group is identified so that the correct version of the form is used.

Form groups let you display specific data for a group of users, or display data in a different way for a group of users. The original Unicenter Service Desk forms are part of the DEFAULT form group. Users get the DEFAULT form group unless the administrator sets up and defines a customization form group for their access type, which means they see these original forms when using the java client.

When modifying an original form, you should look for the form in the DEFAULT form group. You cannot modify the original forms directly using Screen Painter, but you can save a copy of the form under its original name (within its original group) using the Save As option. Within the DEFAULT form group, a modified version of a form always takes precedence over the original version.

You can create new form groups in addition to the DEFAULT form group that is used for the original forms, and save modified versions of the original forms in those form groups. Administrators do this when they want only a specific group of users to use the modified forms. For example, you might want to create versions of some forms that do not include certain features so that you can limit which operations a particular group of users can perform. If you create forms in a form group other than DEFAULT using Screen Painter, you must also define the form group using the java client Form Group Detail window and assign it as the customization form group on the Access Type Detail window for the access type of contacts who will use these customized forms (see the online help for details).

Important! After adding forms to form groups other than DEFAULT, you must recycle the form server to initialize the form group. You need only recycle the form server the first time you add a form to any form group other than DEFAULT.

When deciding which form to use, the java client identifies the customization form group of the user through the access type, and searches that form group for the form name it needs. For example, if the user's access type has no customization form group defined, it searches the DEFAULT form group and uses the form if it is there—within the DEFAULT form group, a modified copy of the form always takes precedence over the original, default version.

If the user's access type has a customization form group that was created specifically for your installation, the java client first searches that form group, then the DEFAULT form group, until the correct form is found. Once again, within the DEFAULT form group, a modified copy of the form still takes precedence over the original, default version.

Form Naming Conventions

Form definitions are stored in the database and retrieved using specific naming conventions. To apply the viewer's default rules to your forms, you must name your forms according to the following conventions, where *xxx* is the same as the object name. The appendix "Objects and Attributes" lists all the objects and attributes that define Unicenter Service Desk:

Form Name	Description
<i>xxx_list</i>	List form
<i>xxx_select</i>	Helper list form
<i>xxx_detail</i>	Detail form
<i>xxx_hier</i>	Hierarchical helper form

How to Customize Forms

The basic steps for customizing a form are:

1. Start Screen Painter.
2. Open the form representing the window you want to modify. If modifying an original form, look for the form in the DEFAULT form group.
3. If the form you want to change is one of the original forms, it will have a Property of Default in the Open Form dialog. You cannot modify the original form directly. Instead, save the form using the Save As menu command, using the same form name. If two forms of the same name exist within the same form group, the modified copy of the form takes precedence over the original, default version.
4. Edit the form, making whatever changes you wish.

Important! If you are creating or updating a form that you want only a specific group of users to view, name your form group something other than DEFAULT. You must also define the form group using the java client Form Group Detail window and assign it as the customization form group on the Access Type Detail window for the access type of contacts who will use these customized forms (see the Screen Painter online help for details).

5. Save the newly edited form, and preview it.
6. Using the java client, choose Form Group, Refresh from the Administration menu.

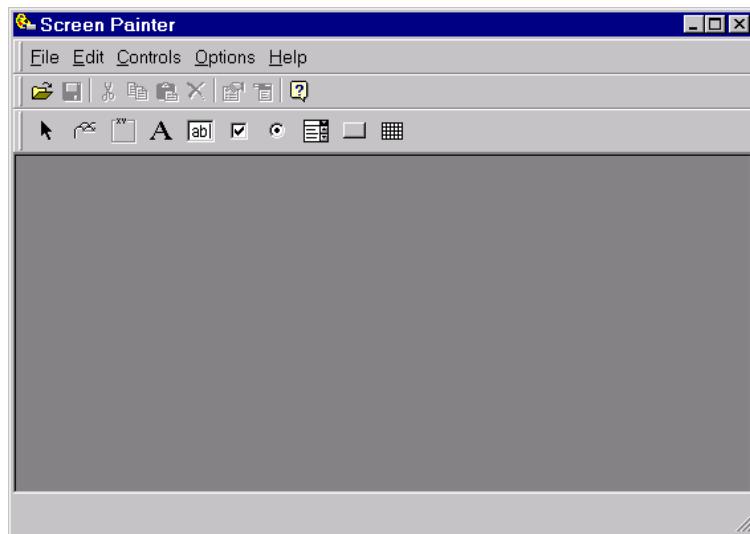
Important! Not all forms can be refreshed using this technique (for example, main_menu). If you refresh the form group and do not see the appropriate changes when you open the corresponding window, you should exit and restart the java client to force the new form to take effect.

7. Test your new form.

Start Screen Painter

Important! Before you can use Screen Painter, you must configure it to work with your database. For more information about instructions, see the chapter "Client Installation (Windows)" in the Implementation Guide. The Unicenter Service Desk daemons (UNIX) or the daemon server (Windows) must also be started on the server machine. For more information about instructions, see the chapter "Servers Management" in the Administrator Guide.

To start Screen Painter, choose Screen Painter from the Unicenter Service Desk menu. After entering your login information, the Screen Painter main window appears, as follows:



Note: To start Screen Painter from the command line, use painter32.exe. See the *CA Reference Guide* for details.

For specific information on how to use Screen Painter once it is running, see the Screen Painter online help.

Controls

Screen Painter cannot be used to add or modify a functionality. Instead, it is used to add or modify GUI controls and then, the functionalities behind the controls are governed by supported back-end logics (majic, spell, or wand).

Add a Control

Adding a control in Screen Painter will not add functionality until a meaningful tag is applied to the control. For example, if you add a text field in a detail form, the text field is enabled, editable and visible in Java Client if you use the default values for Enabled, Locked, and Visible properties of this text field in Screen Painter. However, unless you add a meaningful tag to this text field, the text field will not affect anything in the back store.

Functionality Restrictions

The property of a control does not necessarily reflect the accessibility to a functionality in run time. The functionality provided by a newly added control may be restricted by customized back-end logic (majic, spell, or wand). For example, if a control is added to a form to show a list of contacts and the control's Enabled property is set to true, the control may change to Disabled at run time because the back-end logic (either default or customized) is able to disable the control. Consequently, the functionality to bring up a list of contact is restricted on this control. The same applies for Locked (read only) and Visible properties.

Screen Painter User Access

There is an exception on how back-end logic restricts functionality presented by a control designed in Screen Painter. Users can use Screen Painter to deny access to a functionality provided by a control. If a control is added to a form to show a list of contacts and the control's Enabled property is set to false, the back-end logic (default or customized) is not able to enable it. Consequently, the functionality provided by this control is not accessible by users. This is also valid for Locked and Visible properties.

pdm_manage_user_forms Utility

Service Desk delivers a utility, pdm_manage_user_forms, which allows you to extract, remove, or load java client screen definitions. This utility can help you move forms between your test and production systems. To invoke the utility on Windows, execute the pdm_manage_user_forms command. (On Unix / Linux, execute the pdm_manage_user_forms.pl command instead of pdm_manage_user_forms.) Below are some examples of using the command on a windows system.

To display usage and syntax information, issue the command:
`pdm_manage_user_forms -h`

The following examples detail the usage of this new utility:

To extract a user-modified Call Request detail form from the ANALYST form group, issue the command:

```
pdm_manage_user_forms -e -n cr_detail -g ANALYST -f userform.dat
```

To load that newly created file, issue the command:

```
pdm_manage_user_forms -l -f userform.dat
```

To delete the user-created form from the database, issue the command:

```
pdm_manage_user_forms -r -n cr_detail -g ANALYST
```

Appendix A: Data Element Dictionary

This appendix contains the definitions for the tables in the Unicenter Service Desk database. The tables are found in schema (.sch) files. See the ddict.sch file in the \$NX_ROOT/site directory (UNIX) or *installation-directory*/site directory (Windows) for the most current list of all database tables for your specific installation.

Important! Although several of the tables in this appendix are obsolete, for backward compatibility, or reserved for future use, it is important that you not delete these—or any other table—from the database schema. You can add new tables and add new fields/columns to the existing tables, but never delete any of the tables documented in this appendix. Many applications access the mdb database, so deleting or modifying existing fields or tables could cause some applications to not function properly. Ensure that you follow supported schema modification best practices by using the Web Screen Painter (see page 81).

Access_Levels Table

Access level definitions for Unicenter Service Desk applications.

- **SQL Name**—acc_lvls
- **Object**—acc_lvls

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	Deleted flag: 0=active 1=inactive/marketed as deleted)
enum	INTEGER		Enumerated value for this entry, it specifies ordering in lists and relative values Note: This is a primary key.
id	INTEGER		Unique (to the table) Numeric ID.
nx_desc	nvarchar(40)		This is the description or access level.

[Access_Type Table](#)

Field	Data Type	Reference	Remarks
persid	nvarchar(30)		This is the Persistent ID (SystemObjectName:id).
sym	nvarchar(12)		This is the symbolic name of the level.

[Access_Type Table](#)

Access type information for the Unicenter Service Desk applications.

- **SQL Name**—acctyp
- **Object**—acctyp

Field	Data Type	Reference	Remarks
access_level	INTEGER	Access_Levels ::enum	Enumerated value for this entry, it specifies ordering in lists and relative values. Note: This is a foreign key.
admin	INTEGER		Allows maintenance to administration data.
call_mgr	INTEGER		Permits access to Request Management.
change_mgr	INTEGER		Permits access to Change Order Management.
config	INTEGER		Indicates that it is authorized for the configuration interface.
ct_analyst	INTEGER		Indicates that it is authorized for the CT analyst interface.
ct_customer	INTEGER		Indicates that it is authorized for the CT customer interface.
ct_type	INTEGER		Internal use only

Field	Data Type	Reference	Remarks
default_flag	INTEGER		Specifies the default flag value for this Access Type.
del	INTEGER	Active_Boolean _Table::enum	Indicates the Deleted flag, as follows: 0=Active 1=inactive/marketed as deleted)
description	nvarchar(100)		Describes the textual description of this Access Type
domain	INTEGER	Domain::id	Identifies the unique (to the table) numeric ID. Note: This is a foreign key.
domain_flag	INTEGER		Overrides the Contact's data partition for this Access Type.
external_auth	INTEGER		Allows external authorization for this Access Type.
form_group	INTEGER	Form_Group::id	Identifies the unique (to the table) numeric ID. Note: This is a foreign key.
grant_level	INTEGER	Access_Levels ::enum	Enumerated value for this entry, this specifies ordering in lists and relative values. Note: This is a foreign key.
id	INTEGER		Primary key of this table.
initial_form	nvarchar(256)		The initial form value for this Access Type.

[Access_Type Table](#)

Field	Data Type	Reference	Remarks
interface_type	INTEGER		Identifies the Primary interface type.
inventory	INTEGER		Indicates to allow maintenance to inventory data.
issue_mgr	INTEGER		Permits access to Issue Management.
kcat	INTEGER		Identifies the kcat value for this Access Type.
kd	INTEGER		Specifies the kd value for this Access Type.
kt_admin	INTEGER		Indicates that it is authorized for the KT Admin interface.
kt_analyst	INTEGER		Indicates that it is authorized for the KT Analyst interface.
kt_customer	INTEGER		Indicates that it is authorized for the KT Customer interface.
kt_engineer	INTEGER		Indicates that it is authorized for the KT Engineer interface.
kt_manager	INTEGER		Indicates that it is authorized for the KT Manager interface.
kt_type	INTEGER		Internal use only
last_mod_by	Byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
ldap_access_group	nvarchar(512)		Identifies the ldap access group value for this Access Type.

Field	Data Type	Reference	Remarks
notify	INTEGER		Indicates to allow maintenance to the notification data.
pin_field	nvarchar(50)		Specifies the field that contains the user's pin for pin type authentication.
pref_doc	INTEGER		Defines the Preferred doc, as follows: 1=cr 2=iss
reference	INTEGER		Allows maintenance to reference data.
sd_admin	INTEGER		Indicates that it is authorized for the Service Desk Admin. interface.
sd_analyst	INTEGER		Indicates that it is authorized for the Service Desk Analyst interface.
sd_customer	INTEGER		Indicates that it is authorized for the Service Desk Customer interface.
sd_employee	INTEGER		Indicates that it is authorized for the Service Desk Employee interface.
security	INTEGER		Specifies the security value for this Access Type.
sym	nvarchar(30)		Indicates the symbolic value for this Access Type.
user_auth	INTEGER		Identifies the user authentication type for this Access Type.
view_interna	INTEGER	I	Controls access to internal logs.

Field	Data Type	Reference	Remarks
wsp	INTEGER		Indicates the Web Screen Painter authorization level.

Act_Log Table

Act_Log is a history of activities associated with a call request. The types of activities are listed in the Act_Type table.

- **SQL Name**—act_log
- **Object**—alg

Field	Data Type	Reference	Remarks
action_desc	nvarchar(4000)		Provides the textual description of the activity log entry.
analyst	byte(16)	ca_contact ::uuid	Foreign key to the Contact that created this activity log entry.
call_req_id	nvarchar(30)	Call_Req ::persid	This is the Persistent ID (SystemObjectName:id).
description	nvarchar(4000)		Indicates the textual description of this Activity Log.
id	INTEGER		Identifies the unique (to the table) numeric ID.
internal	INTEGER		Marks this as Internal log.
knowledge _session	nvarchar(80)		Indicates the reference to a knowledge tool session.
knowledge _tool	nvarchar(12)		Specifies the Knowledge tool used for this activity.
last_mod _dt	INTEGER		Indicates the timestamp of when this record was last modified.
persid	STRING 30		Persistent ID (SystemObjectName:id).
system _time	INTEGER		Indicates the Date and Time of record creation.

Field	Data Type	Reference	Remarks
time_spent	INTEGER		Specifies the time spent on this activity by the user.
time_stamp	INTEGER		Specifies the Date and Time of the user activity.
type	nvarchar(12)	Act_Type ::code	(Not Used) Specifies the acknowledgement, which is a noneditable string enum. Note: This is a foreign key.

Act_Type Table

Identifies the activities which may be used to create a history of a ticket. Controls whether the creation of the activity automatically generates a notification to those contacts specified on the call request and what the notification level and message are.

- **SQL Name**—act_type
- **Object**—aty

Note: Adding an activity type requires customization to allow the activity to be part of a ticket's history.

Field	Data Type	Reference	Remarks
chg_default_survey	INTEGER	Survey_Template ::id	Specifies the default survey to use for Change Orders. Note: This is a foreign key.
chg_notify_info	nvarchar(30)	Spell_Macro ::persid	Specifies the information link to the macros for notify.
chg_send_survey	INTEGER		Indicates to send the surveys for Change Orders.
chg_survey_method	INTEGER	Contact_Method ::id	Indicates the notification method to use for Change Order surveys. Note: This is a foreign key.

[Act_Type Table](#)

Field	Data Type	Reference	Remarks
chg_survey_msbody	Long nvarchar		Specifies the Change Order survey message body.
chg_survey_msgtitl	nvarchar(80)		Specifies the Change Order survey message title.
code	nvarchar(12)		This is the primary key of this table.
cr_default_survey	INTEGER	Survey_Template ::id	Indicates the Default survey to use for Requests. Note: This is a foreign key.
cr_notify_info	nvarchar(30)	Spell_Macro::persid	Specifies the information link to the macros for notify.
cr_send_survey	INTEGER		Indicates to send surveys for Requests.
cr_survey_method	INTEGER	Contact_Method ::id	Specifies the notification method to use for Request surveys. Note: This is a foreign key.
cr_survey_msbody	Long nvarchar		Indicates to request the survey message body.
cr_survey_msgtitl	nvarchar(80)		Indicates to request the survey message title.
del	INTEGER	Active_Boolean_Table::enum	Specifies the status of the Deleted flag: 0=Active 1=Inactive/marketed as deleted
description	nvarchar(500)		This is the textual description of this Activity Type.

Field	Data Type	Reference	Remarks
flag1	INTEGER	Boolean_Table ::enum	This is the flag1 value: 1=used by call requests.
flag2	INTEGER	Boolean_Table ::enum	This is the flag2 value: 1=used by change requests.
flag3	INTEGER	Boolean_Table ::enum	This is the flag3 value: 1=Used by issues
flag4	INTEGER	Boolean_Table ::enum	This is the flag4 value: 1=Used by messages
flag5	INTEGER	Boolean_Table ::enum	This is the flag5 value: 1=Used by Knowledge Tools
id	INTEGER		Indicates the numeric ID that is unique to this table.
internal	INTEGER		This is the activity type: 0=User activity 1=Internal
iss_default_survey	INTEGER	Survey_Template ::id	This is the default survey to use for issues. Note: This is a foreign key.
iss_notify_info	nvarchar(30)	Spell_Macro ::persid	Indicates the issue information link to macros for notify.
iss_send_survey	INTEGER		Indicates to send the survey for issues.
iss_survey_method	INTEGER	Contact_Method ::id	Specifies the notification method to use for Issue surveys. Note: This is a foreign key.
iss_survey_msgbody	Long nvarchar		Specifies the Issue survey message body.
iss_survey_msgtitl	nvarchar(80)		Specifies the Issue survey message title.

[Act_Type Table](#)

Field	Data Type	Reference	Remarks
kd_notify_info	nvarchar(30)	Spell_Macro::persid	Indicates the Information link to macros for notify.
last_mod_dt	INTEGER		This is the Timestamp of when this record was last modified.
mgs_notify_info	nvarchar(30)	Spell_Macro::persid	This is the messages Information link to macros for notify.
notify	INTEGER		This specifies whether to generate notifications: 0=No 1=Yes
notify_ack	nvarchar(240)		Indicates the Notification message.
notify_body	Long nvarchar		Indicates the Notification message body.
notify_body_html	Long nvarchar		Specifies the Notification message body html.
notify_level	INTEGER	Notification_Urgency::enum	Specifies the Notification level to use for notifications.
notify_title	nvarchar(80)		Specifies the Notification message title.
persid	nvarchar(30)		Specifies the Persistent ID: (SystemObjectName:id).
sym	nvarchar(30)		Describes the name of the activity.

Act_Type_Assoc Table

Maps a field of an object we want to do activity logging for to an Activity Type (Assignee field on a request is mapped to 'Transfer' Act_Type. All _rsrvd fields are reserved for CA and NOT FOR CLIENT USE)

- **SQL Name**—atyp_asc
- **Object**—act_type_assoc

Field	Data Type	Reference	Remarks
act_type	STRING 30	Act_Type::c ode	Associated activity type
code	STRING 12 UNIQUE NOT_NULL S_KEY		
description	STRING 80		Textual description of this Act Type Association
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
int1_rsrve	INTEGER		Flex field
d			
int2_rsrve	INTEGER		Flex field
d			
int3_rsrve	INTEGER		Flex field
d			
last_mod_ by	UUID	ca_contact: :uuid	UUID of the contact who last modified this record
last_mod_ dt	LOCAL_TIME		Timestamp of when this record was last modified
log_me_f	INTEGER		Logging on/off flag
ob_type	STRING 30		Associated object type
ob_type_at tr	STRING 50		Attribute name within associated object
persid	STRING 30		Persistent ID (SystemObjectName:id)
str1_rsrve	STRING 30		Flex fields - reserved for CA
d			
str2_rsrve	STRING 30		Flex field
d			

Field	Data Type	Reference	Remarks
str3_rsrve d	STRING 30		Flex field
sym	STRING 30 NOT_NULL		

Active_Boolean_Table Table

Table to translate 0 and 1 for the del field on records.

- **SQL Name**—actbool
- **Object**—actbool

Field	Data Type	Reference	Remarks
	INTEGER NOT_NULL		0=active, 1=inactive/marked as deleted
description	STRING 240		Textual description of this boolean value
enum	INTEGER NOT_NULL		Enumerated value for this entry
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
sym	STRING 12 S_KEY		

Active_Reverse_Boolean_Table Table

Table to translate 0 and 1 for the del field on records.

- **SQL Name**—acrtbool
- **Object**—actrbool

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL		0=active, 1=inactive/marked as deleted
description	STRING 240		Textual description of this boolean value
enum	INTEGER NOT_NULL		Enumerated value for this entry
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
sym	STRING 12 S_KEY		

admin_tree Table

Program control table used by Unicenter Service Desk applications.

- **SQL Name**—admin_tree
- **Object**—ADMIN_TREE

Field	Data Type	Reference	Remarks
caption	STRING 50		
description	STRING 255		Textual description of this tree entry
has_children	INTEGER		
id	INTEGER KEY		Unique (to the table) Numeric ID
kt_admin	INTEGER		
kt_ks_caption	STRING 50		

Am_Asset_Map Table

Field	Data Type	Reference	Remarks
kt_ks_flag	INTEGER		
kt_manager	INTEGER		
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
parent_id	INTEGER		
resource	STRING 255		
sd_admin	INTEGER		

Am_Asset_Map Table

Maps a CA Asset Management asset to an internal object am_domain_id and am_unit_id to form a unique AM asset identifier. NOT FOR CLIENT USE.

- **SQL Name**—am_map
- **Object**—am_asset_map

Field	Data Type	Reference	Remarks
am_dmuuid	STRING 64		AM DMUUID of asset
am_domain_id	INTEGER		AM asset domain identifier which created asset
am_server	STRING 64		AM name of UAM domain server
am_type	INTEGER		AM asset type (1=computer; 2=user)
am_unit_domain_id	INTEGER		AM asset domain identifier of asset
am_unit_id	INTEGER		AM asset unit id
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
int1_rsrved	INTEGER		Reserved for CA
int2_rsrved	INTEGER		Reserved for CA
ob_persid	STRING 30		
ob_type	STRING 30		

Field	Data Type	Reference	Remarks
persid	STRING 30		Persistent ID: SystemObjectName:id
str1_rsrved	STRING 80		Reserved for CA
str2_rsrved	STRING 80		Reserved for CA
version	INTEGER NOT_NULL		internal version

Animator Table

Program control table used by Unicenter Service Desk applications

- **SQL Name**—anima
- **Object**—ANI

Field	Data Type	Reference	Remarks
a_act	INTEGER		
a_delta	INTEGER		
a_lock	STRING 200		
a_name	STRING 30 S_KEY		
a_org	LOCAL_TIME		
a_string	STRING 30		
a_time	LOCAL_TIME		
id	INTEGER KEY UNIQUE NOT_NULL		Unique (to the table) Numeric ID
t_method	STRING 30 S_KEY		
t_persid	STRING 30 S_KEY		
t_type	INTEGER		

Archive_Purge_History Table

Historic information about archive/purge activities completed.

- **SQL Name**—arcpur_hist
- **Object**—arcpur_hist

Field	Data Type	Reference	Remarks
chd_obj_deleted	INTEGER		
del	INTEGER NOT_NULL	Active_Boolean_Table::enum	Deleted flag (0=active 1=inactive/marketed as deleted)
end_time	LOCAL_TIME		
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_b	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_d	LOCAL_TIME		Timestamp of when this record was last modified
obj_deleted	INTEGER		
persid	STRING 30		Persistent ID (SystemObjectName:id)
rule_name	STRING 30 NOT_NULL	Archive_Purge_Rule::persid	
start_time	LOCAL_TIME		

Archive_Purge_Rule Table

Rule definitions for the archive/purge engine.

- **SQL Name**—arcpur_rule
- **Object**—arcpur_rule

Field	Data Type	Reference	Remarks
add_query	STRING 240		
arc_file_name	STRING 240		
conf_obj_name	STRING 64		
days_inactive	INTEGER NOT_NULL		
del	INTEGER NOT_NULL	Active_Boolean_Table::enum	Deleted flag (0=active 1=inactive/mark as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_by	UUID	ca_contact::uid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
name	STRING 30 NOT_NULL UNIQUE NOT_NULL S_KEY		Text name of this item
oper_type	INTEGER NOT_NULL		
persid	STRING 30		Persistent ID (SystemObjectName:id)
reoccur_interv	DURATION		
sched	STRING 30	Bop_Workshift ::persid	

Asset_Assignment Table

Describes a relationship between two assets. Each instance of this table is one parent-child arrangement.

- **SQL Name**—hier
- **Object**—hier

Field	Data Type	Reference	Remarks
hier_child	UUID NOT_NULL S_KEY	ca_owned_res child ource::uuid	
hier_license_number	STRING 40		license, serial number
hier_log_date	LOCAL_TIME NOT_NULL		time of assignment
hier_parent	UUID NOT_NULL S_KEY	ca_owned_res parent ource::uuid	
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_by	UUID	ca_contact::u id	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
persid	STRING 30		Persistent ID (SystemObjectName:id)

Atomic_Condition

These define a single condition.

- **SQL Name**—atomic_cond
- **Object**—atomic_cond

Field	Data Type	Reference	Remarks
cond_code	STRING 500		
connecter	INTEGER NOT_NULL		

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL	Active_Boolean_Table::enu m	Deleted flag (0=active 1=inactive/marketed as deleted)
description	STRING 240		Textual description of this condition
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_param	INTEGER NOT_NULL		
last_mod_by	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIM E		Timestamp of when this record was last modified
lval	STRING 30	Act_Type_Assoc::code NOT_NULL	
operator	INTEGER NOT_NULL		
owning_macro	STRING 30	Spell_Macro::persid o	
persid	STRING 30		Persistent ID (SystemObjectName:id)
r_param	INTEGER NOT_NULL		
rval	STRING 50		
rval_assoc	STRING 30	Act_Type_Assoc::code	
sequence	INTEGER NOT_NULL		ordering

Attached_Events Table

This table lists the actual attached events and their information.

- **SQL Name**—att_evt
- **Object**—atev

Field	Data Type	Reference	Remarks
cancel_time	LOCAL_TIME		time canceled
event_tmpl	STRING 30 NOT_NULL S_KEY	Events::persid	Specifies the persid for the attached event.
fire_time	LOCAL_TIME		Datetime event will fire
first_fire_tim e	LOCAL_TIME		Datetime event first fired
group_name	STRING 30		Group smag field
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
num_fire	INTEGER		How many times you fired
obj_id	STRING 30 NOT_NULL		the persid of the object
owning_ast	INTEGER	Attached_SLA: :id	Owning ast object (R11)
persid	STRING 30		Persistent ID (SystemObjectName:id)
start_time	LOCAL_TIME		Datetime event was attached
status_flag	INTEGER		
user_smag	STRING 200		User smag field
wait_time	DURATION		Time to wait before fireing

Attached_SLA Table

Service Level Agreements associated to a ticket.

- **SQL Name**—attached_sla
- **Object**—attached_sla

Field	Data Type	Reference	Remarks
_mapped _chg	INTEGER	chg_id	
_mapped _cr	STRING 30	Call_Req::persid	These are mostly required for advanced runtime queries:
_mapped _iss	STRING 30	issue_persistent_id	
_mapped _iss_wf	INTEGER	isswf::id	
_mapped _wf	INTEGER	wf::id	
del	INTEGER NOT_NULL	Active_Boolean_Ta ble::enum	Deleted flag (0=active 1=inactive/mark as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod _by	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod _dt	LOCAL_TIME		Timestamp of when this record was last modified
last_ttv_ upd	LOCAL_TIME		
map_sds c	STRING 30 NOT_NULL	Service_Desc::cod e	
persid	STRING 30		Persistent ID (SystemObjectName:id)
sla_viol_s tatus	INTEGER		

Field	Data Type	Reference	Remarks
ticket_id	INTEGER NOT_NULL S_KEY		
ticket_ty pe	STRING 30		
time_to_ violation	LOCAL_TIME		
ttv_event	STRING 30	Attached_Events:: persid	

Attachment Table

Object Attachments. (5.0 version)

- **SQL Name**—attmnt
- **Object**—attmnt

Field	Data Type	Reference	Remarks
attmnt_name	STRING 240		attachment name
created_by	UUID	ca_contact::uui d	Who created this attachment
created_dt	LOCAL_TIME		When was this created
del	INTEGER NOT_NULL	Active_Boolean _Table::enum	Deleted flag (0=active 1=inactive/mark as deleted)
description	STRING 500		Textual description of this Attachment
exec_cmd	STRING 12	Remote_Ref::c ode	unix exec string (not currently used)
file_date	LOCAL_TIME		Date of the file.
file_name	STRING 240		server attachment filename
file_size	INTEGER		Size of the file
file_type	STRING 12		File extention (not currently used)

Field	Data Type	Reference	Remarks
folder_id	INTEGER	attmnt_folder::id	folder id
folder_path_ids	STRING 255		folder path ids
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
inherit_permissions_id	INTEGER		The folder ID where pgroup permissions come from
last_mod_by	UUID	ca_contact::uid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
link_only	INTEGER	Boolean_Table:enum	Did we only create a link?
orig_file_name	STRING 240		original file name, URL or UNC
persid	STRING 30		Persistent ID (SystemObjectName:id)
read_pgroup	INTEGER	P_GROUPS::id	Specifies the group of groups eligible to read the document.
rel_file_path	STRING 512		relative path to file
repository	STRING 30	Document_Repository::persid	Which repository
status	STRING 12		Status of Attachment (INSTALLED, INSTALL_FAILED, LINK_ONLY)
write_pgroup	INTEGER	P_GROUPS::id	Specifies the group of groups eligible to update the document.
zip_flag	INTEGER		if the file is zipped

Attachment_Lrel Table

Attachments will eventually have a many to many relationships with CR and Change Orders. Do not alter this table. Fields must be kept in sync with Lrel_Table in bop_schema.sch.

- **SQL Name**—attmnt_lrel
- **Object**—attmnt_lrel

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		left hand attribute name
l_persid	STRING 60		left hand key
l_sql	INTEGER		left hand sort order
r_attr	STRING 30		right hand attribute name
r_persid	STRING 60		right hand key
r_sql	INTEGER		right hand sort order

attmnt_folder Table

List of attachment repository locations.

- **SQL Name**—attmnt_folder
- **Object**—attmnt_folder

Field	Data Type	Reference	Remarks
description	STRING 500		Textual description of this attachment folder
folder_name	STRING 255		folder name
folder_path_	STRING 255		folder path ids ids
folder_type	INTEGER		folder type
has_children	INTEGER		if has childs
id	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID

Field	Data Type	Reference	Remarks
inherit_perm ission_id	INTEGER	attmnt_folder::id	id of folder to inherit from
last_mod_da te	LOCAL_TIME		last modify date
parent_id	INTEGER	attmnt_folder::id	parent folder id
read_pgroup	INTEGER	P_GROUPS::id	Read p Group permissions of the attachment
repository	STRING 30	Document_Repository::persid	repository pers id
write_pgrou p	INTEGER	P_GROUPS::id	Write p Group permissions of the attachment

Attribute_Name Table

Provides corresponding user name for an object attribute. Default population for the table should set at_name and at_dflt to the same thing. The user sees and is able to modify only the at_name. at_dflt is used to restore the default name. at_desc could be user changeable or not. It is of use for identifying the attribute for an as yet unplanned user maintenance application. at_sys should never be seen by the user, nor should at_obj.

- **SQL Name**—atn

Field	Data Type	Reference	Remarks
at_desc	STRING 240		description of attribute
at_dflt	STRING 30		default external name
at_name	STRING 30		user name for attribute
at_obj	STRING 30 S_KEY		system obj name
at_sys	STRING 30 S_KEY		system name
del	INTEGER NOT_NULL		Deleted flag (0=active 1=inactive/mark as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID

Audit_Log Table

Contains all audit log entries.

- **SQL Name**—audit_log
- **Object**—audlog

Field	Data Type	Reference	Remarks
analyst	Byte 16	ca_contact:: uuid	Specifies the user whose update generated this audit record. Note: This is a foreign key.
attr_after_val	nvarchar(160)		Specifies the new value of the object's attribute that has changed.
attr_before_val	nvarchar(160)		Indicates the previous value of the object's attribute that has changed.
attr_name	nvarchar(80)		This is the object's attribute that has changed.
aud_opr	INTEGER		Indicates the type of operation that generated this entry, such as, update, insert, and delete.
audobj_name	nvarchar(10)		Used for tracking the object that has changed.
audobj_persid	nvarchar(30)		Used for tracking the object that has changed.
audobj_trkid	nvarchar(40)		Used for tracking the object that has changed.
audobj_u niqueid	nvarchar(30)		Used for tracking the object that has changed.
change_date	INTEGER		The change date value for this Audit_Log.
id	INTEGER		Specifies the unique (to the table) numeric ID. Note: This is a primary key.

Field	Data Type	Reference	Remarks
int1_rsrved	INTEGER		Reserved
int2_rsrved	INTEGER		Reserved
persid	nvarchar(30)		This is the Persistent ID (SystemObjectName:id).
str1_rsrved	nvarchar(25)		Reserved for future use by CA.

Behavior_Template Table

Each object includes a list of things to execute based on the state of the object. If no behavior is associated with the state transition then it simply controls whether the object can be transitioned to the state.

- **SQL Name**—bhvtpl
- **Object**—bhvtpl

Field	Data Type	Reference	Remarks
context_attrna me	STRING 30		attribute name (eg. state)
context_attrval	INTEGER		attribute value
context_type	STRING 30 NOT_NULL		information which determines whether the transition is valid for the consumercontext object. Note: for general use, the context_attrnamecontext_attrval pair was added. However, can't get the list of valid values very easily from the GUI so we've added an SREL to the object that context_attrnamecontext_attrval really represents. short name of object (eg. wf)
del	INTEGER NOT_NULL	Active_Bool ean_Table:: enum	Deleted flag (0=active 1=inactive/mark as deleted)
description	STRING 500		Textual description of this behavior template

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_by	UUID	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIM E		Timestamp of when this record was last modified
macro_conditio n	STRING 30	Spell_Macro ::persid	controls whether the macro executes
object_id	INTEGER NOT_NULL		id of creator object
object_type	STRING 30 NOT_NULL		short name of creator object
persid	STRING 30		Persistent ID (SystemObjectName:id)
transition_errm sg	STRING 240		text to display on failure
transition_test	STRING 30	Spell_Macro condition ::persid	

Boolean_Table Table

Table of boolean values allows customer to define their text associated with true and false.

- **SQL Name**—bool_tab
- **Object**—bool

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL		Deleted flag (0=active 1=inactive/mark as deleted)
enum	INTEGER NOT_NULL		Enumerated value for this entry - specifies ordering in lists and relative values

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
nx_desc	STRING 40		
sym	STRING 12 UNIQUE NOT_NULL S_KEY		

Bop_Workshift Table

Workshift definition.

- **SQL Name**—bpwshft
- **Object**—wrkshft

Field	Data Type	Reference	Remarks
del	INTEGER	Active_ Boolean_ Table::: enum	This is the Deleted flag: 0=Active 1=Inactive/marketed as deleted
description	nvarchar(80)		This is the textual description of this workshift.
id	INTEGER		Unique (to the table) Numeric ID.
last_mod_by	Byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id). Note: This is a primary key.
sched	nvarchar (1000)		This describes the schedule of this workshift.
sym	nvarchar(30)		Represents the symbolic value for this workshift.

BU_TRANS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—BU_TRANS
- **Object**—BU_TRANS

Field	Data Type	Reference	Remarks
BU_DATE	LOCAL_TIME		
BU_PROCESSE D	INTEGER		
BU_RATING	REAL		
DOC_ID	INTEGER	SKELETONS::id	
HIT_NO_VOTE	INTEGER		
HIT_ORIGIN	INTEGER		
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
INDEX_ID	INTEGER	O_INDEXES::id	
User_slv	INTEGER		
Ticket_slv	INTEGER		
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
USER_ID	UUID	ca_contact::uuid	

Business_Management Table

Reference tables for Change Impact Analyzer Table to hold Service Analyzer relationships between Assets.

- **SQL Name**—busmgt
- **Object**—bmhier

Field	Data Type	Reference	Remarks
bm_rep	INTEGER	busrep id	The Foreign key to the id field of the busrep table.

Field	Data Type	Reference	Remarks
cost	INTEGER		Specifies the cost value for this Change Impact Analyzer relationship.
hier_child	Byte(16)	ca_owned_resource::uuid	Identifies the child asset in this relationship.
hier_parent	Byte(16)	ca_owned_resource::uuid	Identifies the parent asset in this relationship.
id	INTEGER		The Primary key of this table.
last_mod_by	Byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
nx_desc	nvarchar(40)		Specifies the description value for this Business_Management.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sym	nvarchar(60)		Identifies the symbolic value for this Change Impact Analyzer relationship.

Business_Management_Class Table

Table that maps Service Analyzer Classes (as found in NSM COR) to Asset Classes.

- **SQL Name**—buscls
- **Object**—bmcls

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	Identifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted

Field	Data Type	Reference	Remarks
id	INTEGER		This is the primary key to this table.
last_mod_by	Byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
nx_desc	nvarchar(40)		Indicates the description for this Business_Management_Class.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sym	nvarchar(60)		Specifies the symbolic value for this Change Impact Analyzer Class.

Business_Management_Repository Table

Table that keeps track of NSM repositories used by Service Analyzer.

- **SQL Name**—busrep
- **Object**—bmrep

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL	Active_Boolean_Table::enum	Deleted flag (0=active 1=inactive/marketed as deleted)
hostnam e	STRING 64 UNIQUE NOT_NULL		
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_by	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
nx_desc	STRING 40		

Field	Data Type	Reference	Remarks
password	STRING 200		
persid	STRING 30		Persistent ID (SystemObjectName:id)
sym	STRING 60 UNIQUE NOT_NULL S_KEY		
userid	STRING 40		

Business_Management_Repository_Lrel Table

Table that relates Service Analyzer Repositories to Assets.

- **SQL Name**—buslrel
- **Object**—bmlrel

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		
l_persid	STRING 60		
l_sql	INTEGER		
r_attr	STRING 30		
r_persid	STRING 60		
r_sql	INTEGER		

Business_Management_Status Table

Table to hold Service Analyzer Status for Assets.

- **SQL Name**—busstat
- **Object**—bms

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
id	INTEGER		Identifies the unique (to the table) numeric ID.
nx_desc	nvarchar(40)		Identifies the description value for this Business_Management_Status.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id)
status_no	INTEGER		The primary key for this status.
sym	nvarchar(60)		Identifies the symbolic value for this Change Impact Analyzer status.

ca_asset_type Table

Asset types defined in the CA-MDB.

- **SQL Name**—ca_asset_type
- **Object**—ca_asset_type

Field	Data Type	Reference	Remarks
asset_type	STRING 1	_configuration	

Field	Data Type	Reference	Remarks
asset_type	INTEGER		key id
_id	UNIQUE NOT_NULL KEY		
asset_type	STRING 255		
_name			
creation_d	LOCAL_TIME ate		The timestamp indicating when this record was created
creation_u	STRING 64 ser		The name of the person who created this record. Should be in form: LastName, FirstName
delete_tim	LOCAL_TIME e		
exclude_r	INTEGER egistration		
last_updat	LOCAL_TIME e_date		Timestamp of when this record was last modified
last_updat	STRING 64 e_user		Specifies the contact who last modified this record. Should be in form: LastName, FirstName
version_n	INTEGER umber		

ca_company Table

Company information (CA-MDB).

- **SQL Name**—ca_company
- **Object**—ca_cmpny

Field	Data Type	Reference	Remarks
alias	nvarchar(30)		Identifies the Company alias or "also known as" name. For example, CA.
authentication	nvarchar(20) _password		Specifies the authentication password.
authentication	nvarchar(64) _user_name		Specifies the authentication user name.

[ca_company Table](#)

Field	Data Type	Reference	Remarks
bbs	nvarchar(30)		Identifies bulletin board system information.
company_name	nvarchar(100)		Identifies the company name.
company_type	INTEGER	ca_company_type::id	Identifies the Company Type or Vendor Provider. Note: This is the Foreign key to the id field of the ca_company_type table.
company_uuid	Byte(16)		The Primary key of this table.
creation_date	INTEGER		Indicates the date this record was created.
creation_user	nvarchar(64)		Identifies the user or process that created the record.
delete_time	INTEGER		Specifies the delete time.
description	nvarchar(400)		Shows the textual description of this entry.
exclude_registration	INTEGER		Indicates to exclude registration.
inactive	INTEGER	Active_Boolean_Table::enum	Flag representing whether this record is active or inactive. 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Indicates the date that this record was last updated.
last_update_user	nvarchar(64)		Identifies the user or process that last updated the record.
location_uuid	Byte(16)	ca_location::location_uuid	Identifies the location. Note: This is the Foreign key to the ca_location table entry.
month_fiscal_year_ends	INTEGER		Specifies the Integer value (1-12) representing the month the company's fiscal year ends.

Field	Data Type	Reference	Remarks
parent_company_uuid	Byte(16)	ca_company::company_uuid	This is the Foreign key to the company_uuid field of the ca_company table for the company's parent company.
primary_contact_uuid	Byte(16)	ca_contact::uuid	Specifies the Primary contact. Note: This is the Foreign key to the ca_contact table.
source_type_id	INTEGER		Represents the Source type id. Note: This is the Foreign key to the ca_source_type table.
version_number	INTEGER		Specifies the version number for transaction integrity.
web_address	nvarchar(50)		Identifies the company web address.

ca_company_type Table

Company type information (CA-MDB).

- **SQL Name**—ca_company_type
- **Object**—vpt

Field	Data Type	Reference	Remarks
creation_date	INTEGER		Specifies the date this record was created.
creation_user	nvarchar(64)		Identifies the user or process that created the record.
description	nvarchar(255)		Provides the textual company type description of this entry.
id	INTEGER		The Primary key, this also identifies the unique company type numeric ID.

ca_contact Table

Field	Data Type	Reference	Remarks
inactive	INTEGER	Active_Boolean_Table::enum	Specifies Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Specifies the date when this record was last updated.
last_update_user	nvarchar(64)		Specifies the user or process that updated the record.
name	nvarchar(100)		Identifies the unique, company type name.
version_number	INTEGER		Version number for transaction integrity

ca_contact Table

Table of persons that interact with the system in some manner (CA-MDB).

- **SQL Name**—ca_contact
- **Object**—cnt

Field	Data Type	Reference	Remarks
admin_organization_uuid	Byte(16)	ca_organization::organization_uuid	Identifies the Administrative Organization. Note: This is the Foreign key to the ca_organization table.
alias	nvarchar(30)		Identifies the Contact alias or "known as" name.
alt_phone_number	nvarchar(40)		Specifies the alternate phone number.
alternate_identifier	nvarchar(30)		Identifies the alternate contact (for example, the social security number).
comment	nvarchar(255)		Shows the text of the comment.

Field	Data Type	Reference	Remarks
company_uuid	Byte(16)	ca_company ::company_uuid	Identifies the Company. Note: This is the Foreign key to the ca_company table.
contact_type	INTEGER	ca_contact_type::id	Specifies the Contact's type, such as: Customer, Analyst, and so on. Note: This is the Foreign key to the id field of the ca_contact_type table.
contact_uuid	Byte(16)		This is the Primary key, and is a unique identifier.
cost_center	INTEGER	ca_resource_cost_center ::id	Identifies the cost center. Note: This is the Foreign key to the id field of the ca_resource_cost_center table.
creation_date	INTEGER		Identifies the date that the record was created.
creation_user	nvarchar(64)		Specifies the user or process that created this record.
delete_time	INTEGER		Shows the time of deletion.
department	INTEGER	ca_resource_department ::id	Identifies the department. Note: This is the Foreign key to the id field of the ca_resource_department table.
email_address	nvarchar(120)		Identifies the email address.
exclude_registration	INTEGER		Indicates to exclude registration.
fax_number	nvarchar(40)		Identifies the fax number.
first_name	nvarchar(100)		Identifies the first name.

[ca_contact Table](#)

Field	Data Type	Reference	Remarks
floor_location	nvarchar(30)		Identifies the floor location. (for example, the employee works on the first floor).
inactive	INTEGER	Active_Boolean_Table::enum	Specifies the Active flag, as follows: 0=Active 1=Inactive
inrdid	INTEGER		Identifies the role ID.
job_function	INTEGER		Specifies the job function. Note: This is the Foreign key to ca_job_function table.
job_title	INTEGER	ca_job_title :: id	Specifies the job title. Note: This is a Foreign key.
last_name	nvarchar(100)		(Required) Identifies the last name.
last_update_date	INTEGER		Shows the date this record was last updated.
last_update_user	nvarchar(64)		Specifies the user or process that last updated the record.
location_uuid	Byte(16)	ca_location :: location_uuid	Specifies the location. Note: This is the Foreign key to the ca_location table entry.
mail_stop	nvarchar(30)		Identifies the mail stop.
middle_name	nvarchar(100)		Identifies the middle name.
mobile_phone_number	nvarchar(40)		Specifies the mobile phone number.
organization_uuid	Byte(16)	ca_organization :: organization_uuid	Specifies the Organization. Note: This is a Foreign key to the id field of the ca_organization table.

Field	Data Type	Reference	Remarks
pager_email_address	nvarchar(120)		Identifies the pager email address.
pager_number	nvarchar(40)		Identifies the pager number.
pri_phone_number	nvarchar(40)		Identifies the primary phone number.
room_location	nvarchar(30)		Identifies the Room location (for example, the employee works in Cube 123).
supervisor_contact_uuid	Byte(16)	ca_contact ::uuid	Identifies the Supervisor. Note: This is a Foreign key to the ca_contact table.
userid	nvarchar(100)		Specifies the User account id (for example, a user's company employee ID). This ID is unique in order to prevent a user from retrieving another's security settings.
version_number	INTEGER		Version number for transaction integrity.

ca_contact_type Table

Definitions of type/classifications of personal information stored in the ca_contact table (CA-MDB).

- **SQL Name**—ca_contact_type
- **Object**—ctp

Field	Data Type	Reference	Remarks
creation_date	INTEGER		Specifies the date when this record was created.
creation_user	nvarchar(64)		Identifies the user or process that created this record.
delete_time	INTEGER		Specifies the delete time.

[ca_country Table](#)

Field	Data Type	Reference	Remarks
description	nvarchar (255)		Specifies the textual, contact type description.
exclude_registration	INTEGER		Indicates to exclude registration.
hourly_cost	Money		Specifies the hourly cost.
id	INTEGER		Unique (to the table), numeric contact type ID. Note: This is a Primary key.
inactive	INTEGER	Active_Boolean_Table::enum	Identifies the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Specifies the date of when this record was last updated.
last_update_user	nvarchar (64)		Identifies the user or process that last updated the record.
name	nvarchar (100)		Identifies the unique contact type name.
user_uuid	UUID	ca_contact::u uid	This is a unique identifier. Note: This is also the Primary key.
version_number	INTEGER		Version number for transaction integrity.
view_internal	INTEGER		Flag to represent whether this contact type is allowed to view internal data.

[ca_country Table](#)

Country name/code/description information (CA-MDB).

- **SQL Name**—ca_country
- **Object**—country

Field	Data Type	Reference	Remarks
country_code	INTEGER		Identifies the country code ID.

Field	Data Type	Reference	Remarks
creation_date	INTEGER		Specifies the date that the record was created.
creation_user	nvarchar(64)		Specifies the user or process that created the record.
description	nvarchar(255)		Specifies the textual country description.
id	INTEGER		Identifies the unique (to the table) numeric ID. Note: This is a Primary key.
inactive	INTEGER	Active_Boolean_Table::enum	Sets the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Shows the date of when this record was last updated.
last_update_user	nvarchar(64)		Identifies the user or process that last updated the record.
name	varchar(100)		Identifies the unique country name.
phone_mask	nvarchar(60)		Specifies the Phone mask of the given country, for example, (###)###-####x##### ##### #).
version_number	INTEGER		Version number for transaction integrity.

ca_job_function Table

Job function descriptions (CA-MDB).

- **SQL Name**—ca_job_function
- **Object**—job_func

Field	Data Type	Reference	Remarks
creation_date	INTEGER		Specifies the date when this record was created.

ca_job_title Table

Field	Data Type	Reference	Remarks
creation_user	nvarchar(64)		Specifies the user or process that created the record.
delete_time	INTEGER		Identifies the delete time.
description	nvarchar(255)		Specifies the Job function description.
exclude_registration	INTEGER		Indicates to exclude registration.
id	INTEGER		Identifies the unique (to the table) numeric job function ID. Note: This is a Primary key.
inactive	INTEGER	Active_Boolean_Table::enum	Sets the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Specifies the date of when this record was last updated.
last_update_user	varchar(64)		Identifies the user or process that last updated the record.
name	varchar(100)		Identifies the job function name.
version_number	INTEGER		Version number for transaction integrity.

ca_job_title Table

Job title descriptions (CA-MDB).

- **SQL Name**—ca_job_title
- **Object**—position

Field	Data Type	Reference	Remarks
creation_date	INTEGER		Specifies the date of when this record was created.
creation_user	nvarchar(64)		Identifies the user or process that created the record.
delete_time	INTEGER		Indicates the time of deletion.

Field	Data Type	Reference	Remarks
exclude_registration	INTEGER		Indicates to exclude registration.
id	INTEGER		Identifies the unique (to the table) numeric job title ID. Note: This is a Primary key.
inactive	INTEGER	Active_Boolean _Table::enum	Specifies the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Indicates the date of when this record was last updated.
last_update_user	varchar(64)		Specifies the user or process that last updated the record.
name	varchar(100)		Specifies the job title name.
version_number	INTEGER		Version number for transaction integrity.

ca_location Table

Defined locations includes address information for companies and contacts (CA-MDB).

- **SQL Name**—ca_location
- **Object**—loc

Field	Data Type	Reference	Remarks
address_1	nvarchar(50)		This is address line 1.
address_2	nvarchar(50)		This is address line 2.
address_3	nvarchar(50)		This is address line 3.
address_4	nvarchar(50)		This is address line 4.
address_5	nvarchar(50)		This is address line 5.
address_6	nvarchar(50)		This is address line 6.
city	nvarchar(50)		Specifies the name of the city.
comment	nvarchar(255)		Shows the textual comment.

ca_location Table

Field	Data Type	Reference	Remarks
contact_address_flag	INTEGER		Flag indicating that this ca_location record has been created only to specify address information for the contact given in the primary_contact_uuid field.
country	INTEGER	ca_country :: id	Foreign key to the id field of the ca_country table representing the location's country.
county	nvarchar(50)		Identifies the name of the county.
creation_date	INTEGER		Shows the date of when this record was created.
creation_user	nvarchar(64)		Identifies the user or process that created the record.
delete_time	INTEGER		Identifies the time of the deletion.
exclude_registration	INTEGER		Indicates to exclude registration.
fax_number	nvarchar(40)		Identifies the fax number.
geo_coord_type	INTEGER		Foreign key to the id field of the ca_geo_coord_type table for geographic coordinates type (GPS).
geo_coords	nvarchar(40)		Identifies the geographic coordinates values (GPS).
id	INTEGER		Primary key of this table.
inactive	INTEGER	Active_Boolean_Table:: enum	Sets the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Specifies the date the record was last updated.
last_update_user	nvarchar(64)		Identifies the user or process that last updated the record.

Field	Data Type	Reference	Remarks
location_name	nvarchar(100)		Identifies the location name.
location_type_id	INTEGER		Foreign key to the id field of the ca_location_type table representing the location's type.
location_uuid	Byte(16)		This is a unique identifier, and is the Primary key.
mail_address_1	nvarchar(50)		Mailing address 1.
mail_address_2	nvarchar(50)		Mailing address 2.
mail_address_3	nvarchar(50)		Mailing address 3.
mail_address_4	nvarchar(50)		Mailing address 4.
mail_address_5	nvarchar(50)		Mailing address 5.
mail_address_6	nvarchar(50)		Mailing address 6.
parent_location_uuid	byte(16)		Foreign key to the location_uuid field of the ca_location table for the company's parent location.
pri_phone_number	nvarchar(40)		Identifies the private phone number.
primary_contact_uuid	byte(16)		Foreign key to the contact_uuid field of the ca_contact table representing the location's primary contact.
state	INTEGER	ca_state_province::id	Primary key. Identifies the State or Province ID.
version_number	INTEGER		Version number for transaction integrity.
zip	nvarchar(20)		Identifies the zip code.

ca_model_def Table

Model definitions for specific manufacturer/model combinations (CA-MDB).

- **SQL Name**—ca_model_def
- **Object**—mfrmod

Field	Data Type	Reference	Remarks
abbreviation	nvarchar(30)		Specifies the model abbreviation.
capacity	float		Defines the capacity.
capacity_unit	INTEGER		Defines the capacity unit. Note: This is the Foreign key to the id field of the ca_capacity_unit table.
class_id	INTEGER	ca_resource_class::id	Foreign key to the id field of the ca_resource_class table for the class to which this model belongs.
creation_date	INTEGER		Specifies the date of when this record was created.
creation_user	nvarchar(64)		Identifies the user or process that created the record.
current_as_of_date	INTEGER		Specifies the date which represents the point at which the model information is considered current.
delete_time	INTEGER		Indicates the time of deletion.
description	nvarchar(255)		Identifies the model description.
exclude_registration	INTEGER		Indicates to exclude registration.
family_id	INTEGER		Foreign key to the id field of the ca_resource_family table for the family to which this model belongs.
gl_code	INTEGER		Specifies the GL code. Note: This is the Foreign key to the id field of the ca_resource_gl_code table.

Field	Data Type	Reference	Remarks
inactive	INTEGER	Active_Boolean_Table::enum	Specifies the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Specifies the date of when this record was last updated.
last_update_user	nvarchar(64)		Specifies the user or process that last updated this record.
manufacturer_uuid	byte(16)	ca_company::company_uuid	Foreign key to the company_uuid field of the ca_company table for the record that represents the manufacturer.
model_uuid	byte(16)		Specifies the model ID. Note: This is a Primary key.
name	nvarchar(100)		Identifies the unique Model name.
operating_system	INTEGER		Operating system. Note: It is a Foreign key to the id field of the ca_resource_operating_system table.
preferred_seller_uuid	byte(16)		Foreign key to the company_uuid field of the ca_company table for the record that represents the preferred seller company of this model.
subclass_id	INTEGER		Foreign key to the id field of the ca_resource_class table for the subclass to which this model belongs.
version_number	INTEGER		Indicates the version number for transaction integrity.

ca_organization Table

Used to describe a business, or associate businesses with smaller business units, or contacts to businesses.

- **SQL Name**—ca_organization
- **Object**—org

Field	Data Type	Reference	Remarks
abbreviation	nvarchar(30)		Identifies the abbreviation of the organization.
alt_phone_cc	INTEGER		Identifies the alternate phone number country code.
alt_phone_number	nvarchar(32)		Specifies the alternate phone number.
comment	nvarchar (255)		Shows the comment.
company_uuid	byte(16)	ca_company ::company_uuid	Foreign key to the company_uuid field of the ca_company table representing the organization's company.
contact_uuid	byte(16)	ca_contact ::uuid	Foreign key to the contact_uuid field of the ca_contact table representing the organization's primary contact.
cost_center	INTEGER	ca_resource_cost_center::id	Foreign key to the id field of ca_resource_cost_center table which represent this cost center.
creation_date	INTEGER		Identifies the date of when this record was created.
creation_user	nvarchar(64)		Identifies the user or process that created the record.
delete_time	INTEGER		Specifies the time of deletion.
description	nvarchar (255)		Indicates the description of the organization.
email_address	nvarchar (120)		Specifies the email address.

Field	Data Type	Reference	Remarks
exclude_registration	INTEGER		Indicates to exclude registration.
fax_cc	INTEGER		Specifies the fax number's country code.
fax_number	nvarchar(32)		Identifies the fax number.
inactive	INTEGER	Active_Boolean_Table::enum	Specifies the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Specifies the date of when this record was last updated.
last_update_user	nvarchar(64)		Specifies the user or process that last updated this record.
location_uuid	byte(16)	ca_location::location_uuid	Foreign key to the location_uuid field of the ca_location table for the organization's parent location.
org_name	nvarchar(100)		Identifies the name of the organization.
organization_uuid	byte(16)		This is the Primary key for Organization.
pager_email_address	nvarchar(120)		Identifies the pager email address.
parent_org_uuid	byte(16)		Foreign key to the organization_uuid field of the ca_organization table for the organization's parent organization.
pri_phone_cc	INTEGER		Identifies the country code for the primary phone number.
pri_phone_number	nvarchar(32)		Identifies the primary phone number.
version_number	INTEGER		Specifies the version number for transaction integrity.

ca_owned_resource Table

Provides abstract relationship information about an organization's assets such as assignment, ownership, legal documents, maintenance agreements, issues, etc.

- **SQL Name**—ca_owned_resource
- **Object**—nr

Field	Data Type	Reference	Remarks
acquire_date	INTEGER		Specifies the date the resource was acquired.
asset_source_uuid	byte(16)		Foreign key to ca_asset_source table, which establishes the identification of an non-software owned resource to a record in the common asset table through the asset source table.
audit_date	INTEGER		Identifies the audit date.
cabinet_location	nvarchar(30)		Identifies the cabinet location.
company_bought_for_uuid	byte(16)	ca_company::company_uuid	Foreign key to the company_uuid field of the ca_company table representing the company for which this asset was purchased.
cost_center	INTEGER	ca_resource_cost_center::id	Identifies the cost center id. Note: This is a Primary key.
creation_date	INTEGER		Indicates the date when this record was created.
creation_user	nvarchar(64)		Identifies the user or process that created the record.
delete_time	INTEGER		Identifies the date of deletion.
department	INTEGER	ca_resource_department::id	Foreign key to the id field of the ca_resource_department table representing the department to which this asset belongs.

Field	Data Type	Reference	Remarks
dns_name	nvarchar(100)		The name by which this device is known in the domain name server.
exclude_reconciliation	smallint		Flag indicating if the asset is excluded from reconciliation: 0=False 1=True
exclude_registration	INTEGER		Specifies to exclude registration.
floor_location	nvarchar(30)		Identifies the floor location.
gl_code	INTEGER		Foreign key to the id field of the ca_resource_gl_code table.
host_name	nvarchar(255)		Identifies the machine name (hardware only).
inactive	INTEGER	Active_Boolean_Table::enum	Sets the Active flag, as follows: 0=False (Active) 1=True (Inactive)
installation_date	INTEGER		Identifies the date the resource was installed in the organization or network.
ip_address	nvarchar(64)		Identifies the IP address (hardware only).
last_update_date	INTEGER		Specifies the date of when the record was last updated.
last_update_user	nvarchar(64)		Specifies the user or process that last updated this record.
license_information	nvarchar(32)		Identifies the license Information.
license_uuid	byte(16)		Foreign key to ca_license table, which establishes the identification of a software owned resource to a record in the common asset table through the license table.
location_uuid	byte(16)	ca_location ::location_uuid	Foreign key to the location_uuid field of the ca_location table representing the location of this asset.

ca_owned_resource Table

Field	Data Type	Reference	Remarks
mac_address	nvarchar(64)		Identifies the MAC address (hardware only).
maintenance _org_uuid	byte(16)	ca_ organization::organization_uuid	Foreign key to the company_uuid field of the ca_company table representing the vendor responsible for maintenance.
maintenance _vendor_uuid	byte(16)	ca_company::company_uuid	Primary key, this is also a unique identifier.
manufacturer_ uuid	byte(16)	ca_company::company_uuid	Foreign key to the company_uuid field of the ca_company table representing the manufacturer of the asset.
model_uuid	byte(16)	ca_model_def::model_uuid	Foreign key to the model_uuid field of the ca_model_def table to represent the asset model.
operating_ system	INTEGER		Foreign key to the id field of the ca_resource_operating_system table representing the operating system used for this asset.
org_bought_ for_uuid	byte(16)	ca_organization::organization_uuid	Foreign key to the organization_uuid field of the ca_organization table for the organization that bought this asset.
own_resource_ uuid	byte(16)		Primary key. This is the UUID that uniquely identifies the owned resource within tables supporting ownership information.
product_ version	nvarchar(16)		Represents the product version.
purchase_ order_id	nvarchar(20)		Specifies the purchase order ID on which the resource was purchased.

Field	Data Type	Reference	Remarks
requisition_id	nvarchar(50)		Identifies the requisition ID on which the resource was requested.
resource_alias	nvarchar(30)		Identifies the resource alias.
resource_capacity	Float		Specifies the resource capacity, for example, the "20" in 20 MB (applies to hardware only).
resource_capacity_unit	INTEGER		Foreign key to the id field of the ca_capacity_unit table. Specifies the resource capacity unit, for example the "MB" in 20 MB (applies to hardware only).
resource_class	INTEGER	ca_resource_class::id	Primary key that also identifies the class id.
resource_contact_uuid	byte(16)	ca_contact::uuid	Foreign key to the contact_uuid field of the ca_contact table for the primary contact for this asset.
resource_description	nvarchar(255)		Indicates the longer name or description of the resource.
resource_family	INTEGER	ca_resource_family::id	Primary key that is also the resource family id.
resource_name	nvarchar(100)		Identifies the name of the resource.
resource_owner_uuid	byte(16)	ca_contact::uuid	Foreign key to the contact_uuid field of the ca_contact table for the owner of this asset.
resource_quantity	INTEGER		Specifies the resource quantity.
resource_status	INTEGER	ca_resource_status::id	Foreign key to the id field of the ca_resource_status table for resource status.
resource_subclass	INTEGER		Foreign key to the id field of the ca_resource_class table for the resource subclass.

ca_owned_resource Table

Field	Data Type	Reference	Remarks
resource_tag	nvarchar(64)		Specifies the alternate identifier, for example, the alternate id located on the sticker placed on the machine.
responsible_org_uuid	byte(16)	ca_organization::organization_uuid	Foreign key to the organization_uuid field of the ca_organization table for the organization responsible for this asset.
responsible_vendor_uuid	byte(16)	ca_company::company_uuid	Foreign key to the company_uuid field of the ca_company table representing the vendor
room_location	nvarchar(30)		Specifies the room location.
serial_number	nvarchar(64)		Identifies the serial number.
shelf_location	nvarchar(30)		Identifies the shelf location.
slot_location	nvarchar(30)		Specifies the slot location.
status_date	INTEGER		Provides the status date.
supply_vendor_uuid	byte(16)	ca_company::company_uuid	Primary key, which is also a unique identifier.
ufam	smallint		Sets the flag that indicates if the asset is managed by UAPM: 0=False 1=True
version_number	INTEGER		Version number for transaction integrity.

ca_resource_class Table

Definitions of the classifications that may be applied to an asset/resource.

- **SQL Name**—ca_resource_class
- **Object**—grc

Field	Data Type	Reference	Remarks
creation_date	INTEGER		Specifies the date the of when this record was created.
creation_user	nvarchar(64)		Identifies the user or process that created the record.
delete_time	INTEGER		Indicates the date of deletion.
description	nvarchar(255)		Specifies the class description.
exclude_registration	INTEGER		Indicates to exclude registration.
family_id	INTEGER	ca_resource_family::id	Primary key that is also the resource family id.
id	INTEGER		Primary key that is also the class id.
inactive	INTEGER	Active_Boolean_Table::enum	Sets the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Indicates the date the record was last updated.
last_update_user	nvarchar(64)		Specifies the user or process that last updated the record.
name	nvarchar(100)		Identifies the unique class name within family.
parent_id	INTEGER		Foreign key back to the id field of the ca_resource_class table to allow for hierarchical class groupings (for example, subclass).
usp_nsm_class	INTEGER	buscls id	Represents the NSM class. Note: This is the Foreign key to the id field of the buscls table.

ca_resource_cost_center Table

Field	Data Type	Reference	Remarks
version_number	INTEGER		Specifies the version number for transaction integrity.

ca_resource_cost_center Table

An asset is associated with.

- **SQL Name**—ca_resource_cost_center
- **Object**—cost_cntr

Field	Data Type	Reference	Remarks
creation_date	INTEGER		Identifies the date of when this record was created.
creation_user	nvarchar(64)		User or process that created the record.
delete_time	INTEGER		Specifies the time of deletion.
description	nvarchar(255)		Indicates the cost center description.
exclude_registration	INTEGER		Specifies to exclude registration.
id	INTEGER		Primary key that is also the cost center id.
inactive	INTEGER	Active_Boolean_Table::enum	Sets the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Specifies the date of when this record was last updated.
last_update_user	nvarchar(64)		Specifies the user or process that last updated this record.
name	nvarchar(100)		Identifies the unique cost center name.
version_number	INTEGER		Specifies the version number for transaction integrity.

ca_resource_department Table

Department a resource is assigned to/associated with.

- **SQL Name**—ca_resource_department
- **Object**—dept

Field	Data Type	Reference	Remarks
creation_date	INTEGER		Specifies the date of when this record was created.
creation_user	nvarchar(64)		Indicates the user or process that created this record.
delete_time	INTEGER		Indicates the time of deletion.
description	nvarchar (255)		Specifies the department description.
exclude_registration	INTEGER		Specifies to exclude registration.
id	INTEGER		Primary key that is also the resource department id.
inactive	INTEGER	Active_Boolean_Table:: enum	Sets the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Specifies the date of when this record was last updated.
last_update_user	nvarchar(64)		Specifies the user or process that last updated the record.
name	nvarchar (100)		Identifies the unique department name.
version_number	INTEGER		Specifies the version number for transaction integrity.

ca_resource_family Table

High level classification of a resource items, such as Computer, furniture, phone, software and so on.

- **SQL Name**—ca_resource_family
- **Object**—nrf

Field	Data Type	Reference	Remarks
creation_date	INTEGER		Specifies the date of when this record was created.
creation_user	nvarchar(64)		Indicates the user or process that created this record.
delete_time	INTEGER		Indicates the time of deletion.
description	nvarchar(255)		Specifies the family description.
exclude_registration	INTEGER		Specifies to exclude registration.
id	INTEGER		Primary key that is also the resource family id.
inactive	INTEGER	Active_Boolean_Table::enum	Specifies the Active flag, as follows: 0=False (Active) 1=True (Inactive)
include_reconciliation	INTEGER		Specifies the flag that indicates whether to include this family in reconciliation.
last_update_date	INTEGER		Specifies the date of when this record was last updated.
last_update_user	nvarchar(64)		Specifies the user or process that last updated the record.
name	nvarchar(100)		Identifies the unique family name.
table_extension_name	nvarchar(30)		Identifies the name of the associated table to hold extended data for this family.

Field	Data Type	Reference	Remarks
version_number	INTEGER		Indicates the version number for transaction integrity.

ca_resource_gl_code Table

General ledger entry a resource is assigned to/associated with.

- **SQL Name**—ca_resource_gl_code
- **Object**—gl_code

Field	Data Type	Reference	Remarks
creation_date	INTEGER		Specifies the date of when this record was created.
creation_user	nvarchar(64)		Indicates the user or process that created this record.
delete_time	INTEGER		Indicates the time of deletion.
description	nvarchar(255)		Specifies the GL code description.
exclude_registration	INTEGER		Specifies to exclude registration.
id	INTEGER		Primary key that is also the resource GL code id.
inactive	INTEGER	Active_Boolean_Table::enum	Specifies the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Specifies the date of when this record was last updated.
last_update_user	nvarchar(64)		Specifies the user or process that last updated the record.
name	nvarchar(100)		Identifies the unique GL code name.

[ca_resource_operating_system Table](#)

Field	Data Type	Reference	Remarks
version_number	INTEGER		Indicates the version number for transaction integrity.

[ca_resource_operating_system Table](#)

Operating system a resource is assigned to/associated with.

- **SQL Name**—ca_resource_operating_system
- **Object**—opsys

Field	Data Type	Reference	Remarks
creation_date	INTEGER		Specifies the date of when this record was created.
creation_user	nvarchar(64)		Indicates the user or process that created this record.
delete_time	INTEGER		Indicates the time of deletion.
description	nvarchar(255)		Specifies the operating system description.
exclude_registration	INTEGER		Indicates to exclude registration.
id	INTEGER		Primary key that is also the resource operating system id.
inactive	INTEGER	Active_Boolean_Table::enum	Specifies the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Specifies the date of when this record was last updated.
last_update_user	nvarchar(64)		Specifies the user or process that last updated the record.
name	nvarchar(100)		Identifies the unique operating system name.

Field	Data Type	Reference	Remarks
version_number	INTEGER		Specifies the version number for transaction integrity.

ca_resource_status Table

Status of a resource/asset.

- **SQL Name**—ca_resource_status
- **Object**—rss

Field	Data Type	Reference	Remarks
creation_date	INTEGER		Specifies the date of when this record was created.
creation_user	nvarchar(64)		Indicates the user or process that created this record.
delete_time	INTEGER		Indicates the time of deletion.
description	nvarchar(255)		Specifies the operating system description.
exclude_registration	INTEGER		Specifies to exclude registration.
id	INTEGER		Primary key that is also the resource status id.
inactive	INTEGER	Active_Boolean_Table::enum	Specifies the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Specifies the date of when this record was last updated.
last_update_user	nvarchar(64)		Specifies the user or process that last updated the record.
name	nvarchar(100)		Identifies the unique status name.

Field	Data Type	Reference	Remarks
version_number	INTEGER		Indicates the version number for transaction integrity.

ca_site Table

A classification of location information.

- **SQL Name**—ca_site
- **Object**—site

Field	Data Type	Reference	Remarks
alias	nvarchar(30)		Specifies the alias value for this ca_site.
creation_date	INTEGER		Specifies the date of when this record was created.
creation_user	nvarchar(64)		Indicates the user or process that created this record.
delete_time	INTEGER		Indicates the time of deletion.
description	nvarchar(255)		Specifies the ca_site description.
exclude_registration	INTEGER		Specifies to exclude registration.
id	INTEGER		Primary key that is also a unique, numeric ID.
inactive	INTEGER	Active Boolean Table::enum	Specifies the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_update_date	INTEGER		Specifies the date of when this record was last updated.
last_update_user	nvarchar(64)		Specifies the user or process that last updated the record.

Field	Data Type	Reference	Remarks
name	nvarchar (100)		Identifies the unique ca_site name.
version _number	INTEGER		Specifies the version number for transaction integrity.

ca_state_province Table

Sate/Province information associated with a specific locaiton.

- **SQL Name**—ca_state_province
- **Object**—state

Field	Data Type	Reference	Remarks
creation_ date	INTEGER		Specifies the date of when this record was created.
creation _user	nvarchar(64)		Indicates the user or process that created this record.
delete_ time	INTEGER		Indicates the time of deletion.
description	nvarchar (255)		Specifies the state or province name, such as Alberta, Alaska, Alabama, and so on.
exclude_ registration	INTEGER		Specifies to exclude registration.
id	INTEGER		Primary key that is also the unique, state or province ID.
inactive	INTEGER	Active_ Boolean_ Table::enum	Sets the Active flag, as follows: 0=False (Active) 1=True (Inactive)
last_ update_ date	INTEGER		Specifies the date of when this record was last updated.
last_ update _user	nvarchar(64)		Specifies the user or process that last updated the record.

Field	Data Type	Reference	Remarks
symbol	nvarchar(100)		Identifies the state or province symbol, such as AB, AK, AL, and so on..
version_number	INTEGER		Specifies the version number for transaction integrity.

Call_Req Table

Integration with call manager.

- **SQL Name**—call_req
- **Object**—cr

Field	Data Type	Reference	Remarks
active_flag	INTEGER	Boolean_Table::enum	Sets the Active flag, as follows: 0=(Inactive) 1=(Active).
affected_rc	byte(16)	ca_owned_resource ::uuid	Foreign key to the id field of the ca_owned_resource table. It identifies the Asset.
assignee	byte(16)		Foreign key to the contact_uuid field of the ca_contact table. It identifies the Assignee.
call_back_date	INTEGER		Specifies the call back timestamp for this Request.
call_back_flag	INTEGER		Specifies the call back flag value for this Request.
category	nvarchar(30)	Prob_Category::persid	Foreign key to the persistent_id field of the prob_ctg table. This identifies the Category.
change	INTEGER	chg id	Foreign key to the id field of the chg table. This is the Associated Change Order.
charge_back_id	nvarchar(12)		Indicates the user-defined string field.

Field	Data Type	Reference	Remarks
close_date	INTEGER		Represents the timestamp of when this Request was closed.
cr_tticket	INTEGER		Not used.
created_via	INTEGER	Interface::id	Foreign key to the id field of the interface table. Based on site-defined conditions, this reflects which Interface created the request.
customer	byte(16)	ca_contact ::uuid	Foreign key to the contact_uuid field of the ca_contact table. This identifies the Affected End User..
description	nvarchar(4000)		This is the textual description of this call request.
event_token	nvarchar(30)		Used by TNGeh_writer for message matching.
extern_ref	nvarchar(30)		(Deprecated) Specifies the trouble ticket associated with the call request.
group_id	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this represents the Assigned to Group ID.
id	INTEGER		Specifies the unique (to the table) numeric ID
impact	INTEGER	Impact::enum	Foreign key to the enum field of the impact table, this identifies the impact of the Request.
incident_priority	INTEGER		Specifies the computed incident priority if this is an ITIL incident.
last_act_id	nvarchar(12)		Identifies the persid of the last activity.
last_mod_by	byte(16)		Specifies the UUID of the contact who last modified this record.

Call_Req Table

Field	Data Type	Reference	Remarks
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
log_agent	byte(16)	ca_contact ::uuid	Foreign key to the contact_uuid field of the ca_contact table. This identifies who the request was reported by.
macro_predict_viol	INTEGER		Indicates that it is likely to violate its sla (boolean) for action macros to predict sla violations.
open_date	INTEGER		Identifies the timestamp of when the request was created.
parent	nvarchar(30)	Call_Req::persid	Foreign key to the persistent_id field of the call_req table to allow for hierarchical Request groupings (for example, "parent/child").
persid	nvarchar(30)		Identifies the Persistent ID (SystemObjectName:id).
predicted_sla_viol	INTEGER		Indicates that an sla violation has been predicted by Neugents: 1=request
priority	INTEGER	Priority::enum	Foreign key to the enum field of the pri table, this indicates the priority of the call request.
problem	nvarchar(30)		Foreign key to the persistent_id field of the call_req table to allow for linking this incident to a problem.
ref_num	nvarchar(30)		Shows a visible reference number to the user.
resolve_date	INTEGER		Indicates the timestamp of when this Request was resolved.

Field	Data Type	Reference	Remarks
rootcause	INTEGER	Rootcause ::id	Foreign key to the id field of the rootcause table. Specifies the Root cause of the request.
sched_token	nvarchar(128)		Specifies the job scheduling token.
severity	INTEGER	Severity ::enum	Foreign key to the enum field of the sevrtv table, this identifies the severity of the Request.
slaViolation	INTEGER		If defined, this flags the request as follows: Template 1=Request has violated its sla
solution	nvarchar(30)	Call_Req ::persid	(Decprecated) Foreign key to the persistent_id field of the crsol table for old Request solutions.
status	nvarchar(12)	Cr_Status ::code	Foreign key to the code field of the cr_stat table, this is the status of the problem.
string1	nvarchar(40)		Identifies the user defined string field.
string2	nvarchar(40)		Identifies the user defined string field.
string3	nvarchar(40)		Identifies the user defined string field.
string4	nvarchar(40)		Identifies the user defined string field.
string5	nvarchar(40)		Identifies the user defined string field.
string6	nvarchar(40)		Identifies the user defined string field.
summary	nvarchar(240)		Identifies the summary text.
support_lev	nvarchar(30)	Service_Desc::code	Foreign key to the code field of the srv_desc table, this defines the Classic Service Type.

Call_Req_Type Table

Field	Data Type	Reference	Remarks
template_name	nvarchar(30)	Cr_Template::template_name	Foreign key to the template_name field of the cr_tpl table, this specifies the name of the request template.
time_spent_sum	INTEGER		Specifies the sum of activity time spent.
type	nvarchar(10)	crt code	Foreign key to the crt table, this is the Request type.
urgency	INTEGER	Urgency::enum	Foreign key to the enum field of the urgency table, this indicates the call request urgency.

Call_Req_Type Table

Stores codes used in Call_Req.type and the detail form names that should be displayed. This is used by the ITIL vertical market customizations to display alternative forms for cr_detail based on the value of Call_Req.type

- **SQL Name**—crt
- **Object**—crt

Field	Data Type	Reference	Remarks
code	nvarchar(10)Y		Primary key of this table.
del	INTEGER	Active_Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
display_name	nvarchar(30)		The display name value for this Request type.
id	INTEGER		Specifies the unique (to the table) numeric ID.
nx_desc	nvarchar(30)		The description value for this Request type.

Field	Data Type	Reference	Remarks
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sym	nvarchar(30)		The symbolic value for this Request type.

Call_Solution Table

This table exists in the schema for backward compatibility only. Although there is an interface to it, you should not use this table at all; however, it is important that you not delete it from the schema.

- **SQL Name**—crsol
- **Object**—crsol

Field	Data Type	Reference	Remarks
description	STRING 500		Specifies the problem description.
cr_count	INTEGER		Specifies the call request usage count: 0 = Yes 1 = No call request usage count.
cr_flag	INTEGER		Indicates the call request manager flag used by ITIL code.
del	INTEGER NOT_NULL	Active_ Boolean_ Table::enum	Specifies the status of the Deleted flag: 0=Active 1=Inactive/marked as deleted).
id	INTEGER UNIQUE NOT_NULL KEY		Indicates the numeric ID that is unique to the table.
in_flag	INTEGER		Specifies the in_flag used by ITIL code.
last_mod_dt	LOCAL_TIME		Indicates the Timestamp for when this record was last modified.

Change_Act_Log Table

Field	Data Type	Reference	Remarks
nx_desc	STRING 40		Specifies the name for prob_mgr cbox table.
persid	STRING 30		Indicates the Persistent ID: SystemObjectName:id
pr_flag	INTEGER		Specifies the pr_flag used by ITIL code.
sapproved	INTEGER	Boolean_Table::enum	Indicates the status of solution approved.
sname	STRING 40		Specifies the solution name.
solution	STRING 1000		Specifies the solution description.
sym	STRING 60 NOT_NULL S_KEY		Specifies the symbol for the solution.
tcode	INTEGER		This field is no longer used.

Change_Act_Log Table

Change manager tables Change_Act_Log is a history of activities associated with a change request. The types of activities are listed in the Act_Type table.

- **SQL Name**—chgalg
- **Object**—chgalg

Field	Data Type	Reference	Remarks
action_desc	nvarchar(4000)		Shows the text description of the activity log entry.
analyst	byte(16)	ca_contact::uuid	Foreign key to the contact_uuid field of the ca_contact table. This is the Analyst who created this activity log.

Field	Data Type	Reference	Remarks
change_id	INTEGER	chg_id	Foreign key to the id field of the chg table to which the activity log belongs. This is a Change Order.
description	nvarchar(4000)		Textual description of this activity log
id	INTEGER		Unique numeric ID that is the Primary key of this table.
internal	INTEGER		Marks this as 'Internal' log.
knowledge_session	nvarchar(80)		The knowledge session value for this Change_Act_Log.
knowledge_tool	nvarchar(12)		The knowledge tool value for this Change_Act_Log.
last_mod_dt	INTEGER		Specifies the date of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
system_time	INTEGER		Specifies the date and time of record creation.
time_spent	INTEGER		Specifies the time spent by the user on the activity.
time_stamp	INTEGER		Specifies the date and time of activity by the user.
type	nvarchar(12)	Act_Type::code	Identifies the Activity log type. Note: This is the Foreign key to the code field of the act_type table.

Change_Category Table

Change Request categories. Can be hierarchical.

- **SQL Name**—chgcat
- **Object**—chgcat

Field	Data Type	Reference	Remarks
assignee	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is the Assignee.
auto_assign	INTEGER		Flag that enables auto assignment.
cawf_defid	nvarchar(40)		Identifies the CA Workflow definition id.
children_ok	INTEGER		Specifies the handling of the change category: 0=Children not allowed 1=Children are allowed
code	nvarchar(30)		Primary key of this table.
del	INTEGER	Active_Boolean_Table::enum	Delete flag that indicates the following: 0=Active 1=Inactive/mark as deleted
description	nvarchar(500)		Identifies the textual description of this change category.
group_id	byte(16)		Foreign key to the contact_uuid field of the ca_contact table. This is the Group.
id	INTEGER		Specifies the unique (to the table) numeric ID.
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.

Field	Data Type	Reference	Remarks
organization	byte(16)	ca_organization::organization_uuid	Foreign key to the id field of the ca_organization table, this is the Organization.
owning_contract	INTEGER	Service_Contract::id	Foreign key to the id field of the svc_contract table. This is the Service Contract.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id)
schedule	INTEGER		Deprecated.
survey	INTEGER	Survey_Template::id	Foreign key of the id field of the survey_tpl table, this is the Survey.
sym	nvarchar(30)		Change Category symbolic description.

Change_Request Table

Change requests.

- **SQL Name**—chg
- **Object**—chg

Field	Data Type	Reference	Remarks
actions	nvarchar(750)		Identifies the actions value for this Change_Request.
active_flag	INTEGER	Boolean_Table::enum	Flag representing whether this record is active or inactive: 0=Inactive 1=Active
actual_comp_date	INTEGER		Specifies the actual completion date timestamp.
actual_cost	INTEGER		Identifies the actual cost of this Change Order.
actual_total_time	INTEGER		Specifies the sum of actual task time.

Change_Request Table

Field	Data Type	Reference	Remarks
affected_contact	byte(16)	ca_contact ::uuid	Foreign key to the contact_uuid field of the ca_contact table, this specifies the Affected End User.
assignee	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this identifies the Assignee.
backout_plan	nvarchar(4000)		Identifies the backout plan value for this Change_Request.
call_back_date	INTEGER		Identifies the call back timestamp for this Change Order.
call_back_flag	INTEGER		Specifies the call back flag value for this Change Order.
category	nvarchar(12)	Change_Category::id	Foreign key to the code field of the chgctg table, this identifies the Category.
cawf_procid	nvarchar(40)		Identifies the CA Workflow process id.
chg_ref_num	nvarchar(30)		Shows a User visible reference number.
close_date	INTEGER		Shows the timestamp of when the Change Order was closed.
created_via	INTEGER	Interface::id	Foreign key to the id field of the interface table, this identifies which interface created the Change request.
description	nvarchar(4000)		Provides a textual description of this Change Order.
effort	nvarchar(240)		Identifies the effort value for this Change_Request.
est_comp_date	INTEGER		Shows the timestamp for the estimated completion date.

Field	Data Type	Reference	Remarks
est_cost	INTEGER		Specifies the estimated cost of this Change Order.
est_total_time	INTEGER		Identifies the sum of estimated task time.
flag1	INTEGER		This is a user-defined integer flag.
flag2	INTEGER		This is a user-defined integer flag.
flag3	INTEGER		This is a user-defined integer flag.
flag4	INTEGER		This is a user-defined integer flag.
flag5	INTEGER		This is a user-defined integer flag.
flag6	INTEGER		This is a user-defined integer flag.
group_id	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is the Group ID.
id	INTEGER		Primary key of this table, this is a unique numeric ID.
impact	INTEGER	Impact::enum	Foreign key to the enum field of the impact table, this defines the impact.
justification	nvarchar (4000)		Identifies the justification value for this Change_Request.
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
log_agent	byte(16)	ca_contact::uuid	Foreign key to the contact_uuid field of the ca_contact table, this identifies who the change request was reported by.

Change_Request Table

Field	Data Type	Reference	Remarks
macro_predict_viol	INTEGER		Identifies that it is likely to violate its sla (boolean) for action macros to predict sla violations.
need_by	INTEGER		Shows the Need By Date timestamp.
open_date	INTEGER		Shows the timestamp of when this Change Order was created.
organization	byte(16)	ca_organization::organization_uuid	Foreign key to the id field of the ca_organization table, this identifies the Organization.
parent	INTEGER	chg_id	Foreign key to the id field of the chg table to allow for hierarchical Change Order groupings (for example, "parent/child").
persid	nvarchar(30)		Specifies the Persistent ID (SystemObjectName:id).
person_contacting	INTEGER	Person_Contacting::id	Foreign key to the id field of the person table, this identifies the person who made the contact.
predicted_sla_viol	INTEGER		Flag that indicates the following: 1=change order has been predicted by neugents.
priority	INTEGER	Priority::enum	Foreign key to the enum field of the pri table, this identifies the Priority.
product	INTEGER	Product::id	Foreign key to the id field of the product table, this identifies the Product.
reporting_method	INTEGER	Reporting_Method::id	Foreign key to the id field of the repmeth table, this identifies how this Change Order was reported.

Field	Data Type	Reference	Remarks
requestor	byte(16)	ca_contact::uuid	Foreign key to the contact_uuid field of the ca_contact table, this identifies the Requestor.
resolve_date	INTEGER		Indicates the timestamp of when the Change Order was resolved.
rootcause	INTEGER	Rootcause::id	Foreign key to the id field of the rootcause table, this identifies the Root Cause.
service_date	INTEGER		Specifies the service date value for this Change_Request.
service_num	nvarchar(30)		Specifies the service num value for this Change_Request.
slaViolation	INTEGER		Flag indicating the following: 1=change order has violated its sla.
start_date	INTEGER		Specifies the timestamp of when processing started.
status	nvarchar(12)	Change_Status::code	Foreign key to the code field of the chgstat table, this identifies the Status.
string1	nvarchar(40)		This is a user-defined string field.
string2	nvarchar(40)		This is a user-defined string field.
string3	nvarchar(40)		This is a user-defined string field.
string4	nvarchar(40)		This is a user-defined string field.
string5	nvarchar(40)		This is a user-defined string field.
string6	nvarchar(40)		This is a user-defined string field.
summary	nvarchar(240)		Identifies the Change Order summary text.

Field	Data Type	Reference	Remarks
support_level	nvarchar(30)	Service_Desc ::code	Foreign key to the code field of the srv_desc table, this identifies the Classic Service Type.
template_name	nvarchar(30)	chg_template ::template_name	Foreign key to the template_name field of the chg_tpl table, this identifies the Template name.
type_of_contact	INTEGER	Type_Of_Contact::id	Foreign key to the id field of the toc table, this identifies the Type of Contact.
user1	nvarchar(100)		This is a user-defined string field.
user2	nvarchar(100)		This is a user-defined string field.
user3	nvarchar(100)		This is a user-defined string field.

Change_Status Table

Lists the states of the change request, which you can add to at will. This table allows you to control whether the change request is active or inactive when it is changed to this status. Possible statuses include: Open, approval in process, implementation in progress, verification in progress, cancelled, suspended, and closed.

- **SQL Name**—chgstat
- **Object**—chgstat

Field	Data Type	Reference	Remarks
active	INTEGER		Flag that indicates the following: 0=Inactive 1=Active
code	nvarchar(12)		Primary key of this table.
del	INTEGER	Active_Boolean_Table::enum	Delete flag that indicates the following: 0=Active 1=Inactive/marketed as deleted

Field	Data Type	Reference	Remarks
description	nvarchar(500)		Identifies the textual description of this status.
hold	INTEGER		Flag that specifies the following: 0=Start events 1=Stop events
id	INTEGER		Unique (to the table) numeric ID.
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
resolved	INTEGER		Flag that indicates the following: 0=Not yet resolved 1=Resolved
sym	nvarchar(30)		Identifies the Change Request status name.

Chg_Template Table

Maps a template name to a Change_Request that will be used as a template.

- **SQL Name**—chg_template
- **Object**—chg_tpl

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	Delete flag that represents the following: 0=active 1=inactive/mark as deleted
description	nvarchar(1000)		Describes the template.
id	INTEGER		Unique (to the table) numeric ID.

Field	Data Type	Reference	Remarks
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id)
quick_tmpl_type	INTEGER	Quick_Template_Types ::enum	Flag that indicates the following: 1=quick tmpl 2=quick tmpl+review
template	INTEGER	chg id	Unique (to the table) numeric ID.
template_class	nvarchar(12)		Allows subclassing templates.
template_name	nvarchar(30)		Unique name of the template.

Chgcat_Group Table

Used to build a list of Groups that can service the Category.

- **SQL Name**—ccat_grp
- **Object**—chgcat_grp

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		left hand attribute name
l_persid	STRING 60		left hand key
l_sql	INTEGER		left hand sort order
r_attr	STRING 30		right hand attribute name
r_persid	STRING 60		right hand key
r_sql	INTEGER		right hand sort order

Chgcat_Loc Table

Used to build a list of Service Locations valid for the Category.

- **SQL Name**—ccat_loc
- **Object**—chgcat_loc

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		left hand attribute name
l_persid	STRING 60		left hand key
l_sql	INTEGER		left hand sort order
r_attr	STRING 30		right hand attribute name
r_persid	STRING 60		right hand key
r_sql	INTEGER		right hand sort order

Chgcat_Workshift Table

Used to build the list of valid service workshifts for the Category.

- **SQL Name**—ccat_wrkshift
- **Object**—chgcat_workshift

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		left hand attribute name
l_persid	STRING 60		left hand key
l_sql	INTEGER		left hand sort order
r_attr	STRING 30		right hand attribute name
r_persid	STRING 60		right hand key
r_sql	INTEGER		right hand sort order

CI_ACTIONS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—CI_ACTIONS
- **Object**—CI_ACTIONS

Field	Data Type	Reference	Remarks
ACTION_ORDER	INTEGER		
ACTION_TITLE	STRING 100		
ANALYST_ID	UUID	ca_contact:: uuid	
GROUP_ID	UUID	ca_contact:: uuid	
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
PREDEFINED	INTEGER		
STATUS_CURRENT_I D	INTEGER	CI_STATUSE S::id	
UNPUBLISH	INTEGER		
UNRETIRE	INTEGER		
WF_TEMPLATE_ID	INTEGER	CI_WF_TEM PLATES::id	

CI_ACTIONS_ALTERNATE Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—CI_ACTIONS_ALTERNATE
- **Object**—CI_ACTIONS_ALTERNATE

Field	Data Type	Reference	Remarks
ACTION_ID	INTEGER	CI_ACTIONS::id	

Field	Data Type	Reference	Remarks
CONTACT_ID	UUID	ca_contact::uuid	
CONTACT_TYPE	INTEGER		
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified

CI_BOOKMARKS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—CI_BOOKMARKS
- **Object**—CI_BOOKMARKS

Field	Data Type	Reference	Remarks
BOOKMARK_TITLE	STRING 100		
DOCUMENT_ID	INTEGER	SKELETON S:::id	
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
USER_ID	UUID	ca_contact: :uuid	

CI_DOC_LINKS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—CI_DOC_LINKS
- **Object**—CI_DOC_LINKS

Field	Data Type	Reference	Remarks
DOC_ID1	INTEGER	SKELETONS ::id	
DOC_ID2	INTEGER	SKELETONS ::id	
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
LAST_MO_D_DT	LOCAL_TIME		Timestamp of when this record was last modified

CI_DOC_TEMPLATES Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—CI_DOC_TEMPLATES
- **Object**—CI_DOC_TEMPLATES

Field	Data Type	Reference	Remarks
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
IS_DEFAULT	INTEGER		
IS_PREDEFINE_D	INTEGER		
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
PAGE_HTML	STRING 32768		
TEMPLATE_NA_ME	STRING 255		

CI_DOC_TYPES Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—CI_DOC_TYPES
- **Object**—CI_DOC_TYPES

Field	Data Type	Reference	Remarks
DOC_TYPE_TXT	STRING 50		
ID	INTEGER KEY	Unique (to the table) Numeric ID	
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified

CI_PRIORITIES Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—CI_PRIORITIES
- **Object**—CI_PRIORITIES

Field	Data Type	Reference	Remarks
ID	INTEGER KEY	Unique (to the table)	Numeric ID
LAST_MO D_DT	LOCAL_TIME		Timestamp of when this record was last modified
PRIORITY	STRING 50		

CI_STATUSES Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—CI_STATUSES
- **Object**—CI_STATUSES

Field	Data Type	Reference	Remarks
ID	INTEGER KEY	Unique (to the table)	Numeric ID

[CI_WF_TEMPLATES](#) Tables

Field	Data Type	Reference	Remarks
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
PREDEFINED	INTEGER		
STATUS	STRING 50		
STATUS_DESC	STRING 255		
RIPTION			
STATUS_ORDER	INTEGER		
R			

[CI_WF_TEMPLATES](#) Tables

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—CI_WF_TEMPLATES
- **Object**—CI_WF_TEMPLATES

Field	Data Type	Reference	Remarks
ID	INTEGER KEY		Unique (to the table) Numeric ID
IS_DEFAULT	INTEGER		
LAST_MOD_DATE	LOCAL_TIME		Timestamp of when this record was last modified
WF_DESCRIPTION	STRING 255		
WF_NAME	STRING 255		

Column_Name Table

Column Name list used by Web Screen Painter.

- **SQL Name**—cn

Field	Data Type	Reference	Remarks
cn_desc	STRING 240		description of column
cn_dflt	STRING 30		default external name

Field	Data Type	Reference	Remarks
cn_name	STRING 30		user name for column
cn_sys	STRING 30 S_KEY		system name
cn_table	STRING 30 S_KEY		system table name
del	INTEGER NOT_NULL		Deleted flag (0=active 1=inactive/mark as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID

Contact_Method Table

Defines contact method types. The cm_template field is a command string that gets executed as a script (with environment variables set) by the notify subsystem.

- **SQL Name**—ct_mth
- **Object**—cmth

Field	Data Type	Reference	Remarks
cm_template	nvarchar(240)		Specifies the method template.
del	INTEGER	Active_Boolean_Table::enum	Identifies the Delete flag, as follows: 0=Active 1=Inactive/mark as deleted)
id	INTEGER		Primary key of this table, that is also a unique, numeric ID.
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the Timestamp of when this record was last modified.
nx_desc	nvarchar(40)		Specifies the Contact method description.

Field	Data Type	Reference	Remarks
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sym	nvarchar(60)		Identifies the Contact method symbolic name.
write_file	INTEGER		Flag that indicates the following: 1=write output to the file

Controlled_Table Table

Program control table used by Unicenter Service Desk applications.

- **SQL Name**—ctab
- **Object**—ctab

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean _Table::enum	Indicates the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted)
id	INTEGER		Primary key to this table, it is a unique numeric ID.
nx_desc	nvarchar(40)		Specifies the Table description.
obj_name	nvarchar(30)		Specifies the Majic object name that corresponds to this table.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sym	nvarchar(30)		Represents the symbolic name of this controlled table.

Cr_Call_Timers Table

Call Request call timers. A stop watch with various thresholds that gives the help desk analyst a visual and audio indication of elapsed time.

- **SQL Name**—crctmr
- **Object**—ctimer

Field	Data Type	Reference	Remarks
beep	INTEGER		A beep indicator for when the threshold is reached: 0=no beep 1=beep
color	STRING 12		Indicates the color of the timer at the start time.
del	INTEGER NOT_NULL	Active_ Boolean_ Table::enum	Specifies the status of the Delete flag: (0=active 1=inactive/marked as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		The ID unique (to the table) Numeric ID.
last_mod_by	UUID	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	LOCAL_TIME		Indicates the timestamp for when this record was last modified.
persid	STRING 30		Identifies the Persistent ID (SystemObjectName:id).
text	STRING 240		Identifies the threshold text to display when the timer indicates elapsed time.
threshold	DURATION NOT_NULL		Identifies the threshold elapsed time.

Cr_Status Table

Call Request Status. Lists the states of the call request. May be added to at will. Allows the user to control whether the call request is active or inactive when it is changed to this status.

- **SQL Name**—cr_stats
- **Object**—crs

Field	Data Type	Reference	Remarks
active	INTEGER		Sets the Active flag, as follows: 0=Inactive 1=Active
code	nvarchar(12)		Primary key of this table.
cr_flag	INTEGER		When this flag is set to 1, this status is valid for Requests.
del	INTEGER	Active_ Boolean_ Table::: enum	The Delete flag that indicates the following: 0=Active 1=Inactive/marketed as deleted
description	nvarchar(500)		Identifies the textual description of the status.
hold	INTEGER		Sets the Hold flag, as follows: 0=Start events 1=Stop events
id	INTEGER		Unique (to the table) numeric ID.
in_flag	INTEGER		When this flag is set to 1, the status is valid for Incidents.
last_mod_by	byte(16)	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
pr_flag	INTEGER		When this flag set to 1, the status is valid for Problems.

Field	Data Type	Reference	Remarks
resolved	INTEGER		Flag that indicates the following: 0= Not yet resolved 1=Resolved
sym	nvarchar(30)		Identifies the symbol of the Request status name.

Cr_Stored_Queries Table

Custom bin stored queries. System administrators may add to this table at will. Determines which queries may be used by help desk analysts to customize their scoreboard.

- **SQL Name**—crsq
- **Object**—crsq

Field	Data Type	Reference	Remarks
description	STRING 240		The textual description of this stored query.
code	STRING 12 UNIQUE NOT_NULL S_KEY		The non-editable handle for the query.
count_url	STRING 240		The URL for counts, if obj_type=url.
criteria	STRING 240		The where clause for querying.
del	INTEGER NOT_NULL	Active_Boolean_Table::enum	The deleted flag (0=active 1=inactive/mark as deleted).
id	INTEGER UNIQUE NOT_NULL KEY		The unique (to the table) Numeric ID.
label	STRING 80		The label to display on the scoreboard.
last_mod_by	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.

Cr_Template Table

Field	Data Type	Reference	Remarks
last_mod_dt	LOCAL_TIME		The Timestamp for when this record was last modified.
obj_type	STRING 30		The scoreboard, with the capability of having enough space to allow for expansions to accommodate the cr, tt, ir, and chg types.
persid	STRING 30		The Persistent ID (SystemObjectName:id).

Cr_Template Table

Request Template Table maps a template name to a Call_Req that will be used as a template.

- **SQL Name**—cr_template
- **Object**—cr_tpl

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	The Delete flag that indicates the following: 0=Active 1=Inactive/marketed as deleted)
description	nvarchar(1000)		Shows the description of the template.
id	INTEGER		Identifies the unique (to the table) numeric ID.
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
quick_tmpl_type	INTEGER	Quick_Template_Types::enum	Flag that indicates the following: 1=Quick tmpl 2=Quick tmpl+review

Field	Data Type	Reference	Remarks
template	nvarchar(30)	Call_Req:: persid	Persistent ID (SystemObjectName:id).
template _class	nvarchar(12)		Indicates to allow subclassed templates.
template _name	nvarchar(30)		Identifies the unique name of the template.

D_PAINTER Table

Tables to be used for Form Server and Screen Painter For the new GUI.

- **SQL Name—D_PAINTER**

Field	Data Type	Reference	Remarks
CNTLID	INTEGER		id of the control
CNTLTYPE	INTEGER		type of control
DDID	INTEGER		data dictionary id
ENTITYID	INTEGER		entity type
EXTRA_L1	INTEGER		user-definable
EXTRA_L2	INTEGER		user-definable
EXTRA_L3	INTEGER		user-definable
EXTRA_S1	STRING 50		user-definable
EXTRA_S2	STRING 50		user-definable
EXTRA_S3	STRING 50		user-definable
FORMGROUP	STRING 50		group in which the form is contained
FORMID	INTEGER		id number of the form
FORMNAME	STRING 50		name of the form
FORMTYPE	INTEGER		type of form
ID	INTEGER UNIQUE NOT_NULL KEY		key ID
MAPBACK	STRING 30		data dictionary owner

Field	Data Type	Reference	Remarks
PARENTID	INTEGER		control id of parent control
PREDEFINED	INTEGER		0=normal screen, 2=default screen
PROPLIST	STRING 1000		properties for the control
SECLEVEL	INTEGER		security level
TSTAMP	REAL		time stamp

Delegation_Server Table

List of servers that can be delegated from this one along with xport methods.

- **SQL Name**—dlgsrvr
- **Object**—dlgsrvr

Field	Data Type	Reference	Remarks
anon_userid	STRING 8		Anonymous userid
appl_addr	STRING 48		Name or address of application
default_assignee	UUID	ca_contact:: uuid	Assignee for incoming tickets
default_userid	STRING 8		Default userid
del	INTEGER NOT_NULL	Active_Bool ean_Table:: enum	Deleted flag (0=active 1=inactive/marketed as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
metafile	STRING 256		path to conversion metafile def
nx_desc	STRING 40		Description
password	STRING 16		Server password
server	STRING 128		Server name or ip address
sym	STRING 64 UNIQUE NOT_NULL S_KEY		System name

Field	Data Type	Reference	Remarks
transport	INTEGER		

Document_Repository Table

Contains Information on document repositories, which are used to store attachments.

- **SQL Name**—doc_rep
- **Object**—doc_rep

Field	Data Type	Reference	Remarks
description	STRING 500		description
archive_pat	STRING h 255		
archive_typ	INTEGER e		
cgi_path	STRING 255		location and name of CGI
default_rep	INTEGER		
del	INTEGER NOT_NULL	Active_Bool ean_Table:: enum	Deleted flag (0=active 1=inactive/marketed as deleted)
file_limit_si	INTEGER ze		file limit size
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_b	UUID y	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record
last_mod_d	LOCAL_TIM t	E	Timestamp of when this record was last modified
persid	STRING 30		Persistent ID (SystemObjectName:id)
prohibited_ext	STRING 500		Prohibited file extensions
protocol	STRING 12		HTTP or SHARE

Field	Data Type	Reference	Remarks
repository_type	INTEGER		type of repository (attachments, knowledge)
retrieve_path	STRING 255		How to get back to upload_path via protocol
server	STRING 30		Name of Doc Server
servlet_path	STRING 255		servlet URL
sym	STRING 30 NOT_NULL S_KEY		name of document repository
upload_path	STRING 255		Server Location of doc repository

Domain Table

Lists the names and descriptions of the data partitions themselves.

- **SQL Name**—dmn
- **Object**—dmn

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
id	INTEGER		Primary key of this table, it is a unique numeric ID.
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
nx_desc	nvarchar(40)		Identifies the data partition description.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).

Field	Data Type	Reference	Remarks
sym	nvarchar(30)		Identifies the Data partition name.

Domain_Constraint Table

Lists the constraints associated with a particular data partition and controlled table.

- **SQL Name**—dcon
- **Object**—dcon

Field	Data Type	Reference	Remarks
alias	INTEGER		Identifies the alias value for this Domain_Constraint.
constraint _majic	nvarchar(4000)		Specifies Constraint (Majic).
constraint _SQL	nvarchar(4000)		Specifies Constraint (SQL).
del	INTEGER	Active_ Boolean_ Table::enum	Indicates the Delete flag, as follows: 0=Active 1=inactive/mark as deleted)
dom_id	INTEGER	Domain::id	Identifies the unique (to the table) numeric ID.
error_msg	nvarchar(150)		Specifies the message on violation.
id	INTEGER		Primary key to this table, it is a unique numeric ID.
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
tbl_id	INTEGER	ctab::id	Indicates the unique (to the table) numeric ID.

Domain_Constraint_Type Table

Field	Data Type	Reference	Remarks
type	INTEGER	Domain_Constraint_Type::enum	Enumerated value for this entry, this specifies ordering in lists and relative values.

Domain_Constraint_Type Table

Lists the types of constraints that can be associated with a particular data partition and controlled table.

- **SQL Name**—dcon_typ
- **Object**—dcon_typ

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/mark as deleted
enum	INTEGER		Enumerated value for this entry, this specifies ordering in lists and relative values.
id	INTEGER		Indicates the unique (to the table) numeric ID.
nx_desc	nvarchar(40)		Specifies the description of the domain constraint type.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sym	nvarchar(12)		Identifies the symbolic name of this constraint type.

EBR_ACRONYMS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_ACRONYMS
- **Object**—EBR_ACRONYMS

Field	Data Type	Reference	Remarks
ACRONYM	STRING 50		
ID	INTEGER NOT_NULL KEY	Unique (to the table) Numeric ID	
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified

EBR_DICTIONARY Table

- **SQL Name**—EBR_DICTIONARY
- **Object**—EBR_DICTIONARY

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE KEY		
WORD_ID	INTEGER		
WORD	STRING 50 NOT_NULL S_KEY		
WORD_TYPE	INTEGER		
WORD_TOTAL_COUNT	INTEGER		
DF	INTEGER		
WORD_IDF	INTEGER		
last_mod_dt	LOCAL_TIME		

EBR_DICTIONARY_ADM Table

- **SQL Name**—EBR_DICTIONARY_ADM
- **Object**—EBR_DICTIONARY_ADM

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE KEY		
WORD_ID	INTEGER		
WORD	STRING 50 NOT_NULL S_KEY		
WORD_TYPE	INTEGER		
WORD_TOTAL_COUNT	INTEGER		
DF	INTEGER		
WORD_IDF	INTEGER		
last_mod_dt	LOCAL_TIME		

EBR_FULLTEXT Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_FULLTEXT
- **Object**—EBR_FULLTEXT

Field	Data Type	Reference	Remarks
DOC_TYPE	INTEGER		
ENTITY_ID	INTEGER		
FULL_WORD	STRING 50		
FULL_WORD_REVERSE	STRING 50		
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
PERMISSION_INDEX_I	INTEGER		
D			
PRODUCT	STRING 50		

Field	Data Type	Reference	Remarks
SHORT_WORD	STRING 50		
TABLE_ID	INTEGER		
WORD_COUNT	INTEGER		
WORD_COUNT_PROBL EM	INTEGER		
WORD_COUNT_RESOL UTION	INTEGER		
WORD_COUNT_SUMMA RY	INTEGER		
WORD_COUNT_TITLE	INTEGER		
WORD_IDF	INTEGER		
WORD_ORDER	INTEGER		
WORD_TYPE	INTEGER		

EBR_FULLTEXT_ADMIN Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_FULLTEXT_ADMIN
- **Object**—EBR_FULLTEXT_ADMIN

Field	Data Type	Reference	Remarks
DOC_TYPE	INTEGER		
ENTITY_ID	INTEGER		
FULL_WORD	STRING 50		
FULL_WORD_	STRING 50		
	REVERSE		
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
PERMISSION	INTEGER		
_INDEX_ID			
PRODUCT	STRING 50		
SHORT_WOR	STRING 50		
D			

EBR_FULLTEXT_SD Table

Field	Data Type	Reference	Remarks
TABLE_ID	INTEGER		
WORD_COUN	INTEGER		T
WORD_COUN	INTEGER		T_PROBLEM
WORD_COUN	INTEGER		T_RESOLUTI
		ON	
WORD_COUN	INTEGER		T_SUMMARY
WORD_COUN	INTEGER		T_TITLE
WORD_IDF	INTEGER		
WORD_ORDE	INTEGER		R
WORD_TYPE	INTEGER		

EBR_FULLTEXT_SD Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_FULLTEXT_SD
- **Object**—EBR_FULLTEXT_SD

Field	Data Type	Reference	Remarks
ENTITY_ID	INTEGER		
FULL_WORD	INTEGER		
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
SHORT_WORD	STRING 50		
TABLE_ID	INTEGER		
WORD_COUNT	INTEGER		
WORD_COUNT_PROBLE	INTEGER	M	

Field	Data Type	Reference	Remarks
WORD_COUNT_RESOLUTION	INTEGER		
WORD_COUNT_SUMMARY	INTEGER		
WORD_COUNT_TITLE	INTEGER		
WORD_IDF	INTEGER		
WORD_ORDER	INTEGER		
WORD_TYPE	INTEGER		

EBR_FULLTEXT_SD_ADM Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_FULLTEXT_SD_ADM
- **Object**—EBR_FULLTEXT_SD_ADM

Field	Data Type	Reference	Remarks
ENTITY_ID	INTEGER		
FULL_WORD	INTEGER		
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
SHORT_WORD	STRING 50		
TABLE_ID	INTEGER		
WORD_COUNT	INTEGER		
WORD_COUNT_PROBLEM	INTEGER		
WORD_COUNT_RESOLUTION	INTEGER		
WORD_COUNT_SUMMARY	INTEGER		
WORD_COUNT_TITLE	INTEGER		
WORD_IDF	INTEGER		
WORD_ORDER	INTEGER		

EBR_INDEX Table

Field	Data Type	Reference	Remarks
WORD_TYPE	INTEGER		

EBR_INDEX Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_INDEX
- **Object**—EBR_INDEX

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE KEY		
ENTITY_ID	INTEGER NOT_NULL S_KEY		
WORD_ID	INTEGER NOT_NULL S_KEY		
WORD_TYPE	INTEGER NOT_NULL S_KEY		
WORD_ORDER	INTEGER		
WORD_COUNT	INTEGER		
WORD_COUNT_TITLE	INTEGER		
WORD_COUNT_SUMMAR Y	INTEGER		
WORD_COUNT_PROBLEM	INTEGER		
WORD_COUNT_RESOLUT ION	INTEGER		

EBR_INDEX_ADMIN Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_INDEX_ADMIN
- **Object**—EBR_INDEX_ADMIN

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE KEY		
ENTITY_ID	INTEGER NOT_NULL S_KEY		
WORD_ID	INTEGER NOT_NULL S_KEY		
WORD_TYPE	INTEGER NOT_NULL S_KEY		
WORD_ORDER	INTEGER		
WORD_COUNT	INTEGER		
WORD_COUNT_TITLE	INTEGER		
WORD_COUNT_SUMMAR Y	INTEGER		
WORD_COUNT_PROBLEM	INTEGER		
WORD_COUNT_RESOLUT ION	INTEGER		

EBR_INDEXING_QUEUE Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_INDEXING_QUEUE
- **Object**—EBR_INDEXING_QUEUE

Field	Data Type	Reference	Remarks
ACTION	INTEGER		
ACTION_DATE	DATE		

EBR_KEYWORDS Table

Field	Data Type	Reference	Remarks
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
INDEXED	INTEGER		
OBJ_PERSID	STRING 30		Persistent ID (SystemObjectName:id)
PRIORITY	INTEGER		
TEXT	STRING 32768		
TEXT	STRING 32768		

EBR_KEYWORDS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_KEYWORDS
- **Object**—EBR_KEYWORDS

Field	Data Type	Reference	Remarks
ENTITY_ID	INTEGER		
EXT_TABLE_ID	INTEGER		
FULL_WORD	STRING 50		
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID

EBR_LOG Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_LOG
- **Object**—EBR_LOG

Field	Data Type	Reference	Remarks
ASKED_DATE	LOCAL_TIME		
BEST_IDS	STRING 110		
EXTERNAL_ID	STRING 50		

Field	Data Type	Reference	Remarks
FILTER_DATA	STRING 32768		
FUZZINESS	INTEGER		
ID	INTEGER KEY		Unique (to the table) Numeric ID
KEYWORDS	STRING 32768		
MATCH_TYPE	INTEGER		
METHOD_PERFORM ANCE	INTEGER		
METHOD_TYPE	INTEGER		
NUM_MATCHES	INTEGER		
ORDER_DIRECTION	INTEGER		
PRIMARY_ORDER	STRING 50		
ROWS_FOUND	INTEGER		
ROWS_TO_FETCH	INTEGER		
SEARCH_IN	INTEGER		
SEARCH_QUALITY	INTEGER		
SEARCH_TEXT	STRING 255		
SEARCH_TYPE	INTEGER		
SECONDARY_ORDE R	INTEGER		
SESSION_ID	INTEGER		
SQL_TEXT	STRING 32768		
TOP_MATCH_ID	INTEGER		
UNIQUE_WORD_CO UNT	INTEGER		
USER_ID	STRING 100		
WORD_COUNT	INTEGER		

EBR_METRICS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_METRICS
- **Object**—EBR_METRICS

Field	Data Type	Reference	Remarks
COMMENTS	STRING 255		
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
METRIC	STRING 50		
WEIGHT	REAL		

EBR_NOISE_WORDS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_NOISE_WORDS
- **Object**—EBR_NOISE_WORDS

Field	Data Type	Reference	Remarks
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
NOISE_WORD	STRING 50		

EBR_PATTERNS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_PATTERNS
- **Object**—EBR_PATTERNS

Field	Data Type	Reference	Remarks
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
PATTERN_DEFAULT	STRING 255		
PATTERN_NAME	STRING 50		
PATTERN_VALUE	STRING 255		
PATTERN_VALUE_ADM	STRING 255		

EBR_PREFIXES Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_PREFIXES
- **Object**—EBR_PREFIXES

Field	Data Type	Reference	Remarks
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
PREFIX	STRING 50		

EBR_PROPERTIES Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_PROPERTIES
- **Object**—EBR_PROPERTIES

Field	Data Type	Reference	Remarks
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
PROPERTY_ADMIN	INTEGER		
PROPERTY_DEFAULT	STRING 50		
PROPERTY_NAME	STRING 50 S_KEY		
PROPERTY_TYPE	STRING 50		
PROPERTY_VALUE	STRING 32768		
PROPERTY_VALUE_ADM	STRING 32768		

EBR_SUBSTITS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_SUBSTITS
- **Object**—EBR_SUBSTITS

Field	Data Type	Reference	Remarks
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
SYMBOL1	STRING 50		
SYMBOL2	STRING 50		

EBR_SUFFIXES Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_SUFFIXES
- **Object**—EBR_SUFFIXES

Field	Data Type	Reference	Remarks
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
SUFFIX	STRING 50		

EBR_SYNONYMS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_SYNONYMS
- **Object**—EBR_SYNONYMS

Field	Data Type	Reference	Remarks
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
KEYWORD1	STRING 50		
KEYWORD2	STRING 50		
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified

EBR_SYNONYMS_ADM Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—EBR_SYNONYMS_ADM
- **Object**—EBR_SYNONYMS_ADM

Field	Data Type	Reference	Remarks
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID

[ES_CONSTANTS Table](#)

Field	Data Type	Reference	Remarks
KEYWORD1	STRING 50		
KEYWORD2	STRING 50		
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified

[ES_CONSTANTS Table](#)

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—ES_CONSTANTS
- **Object**—ES_CONSTANTS

Field	Data Type	Reference	Remarks
COMMENTS	STRING 255		
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
NAME	STRING 50		Text name of this item
PROPERTYID	INTEGER		
PROPVVALUE	INTEGER		

[ES_NODES Table](#)

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—ES_NODES
- **Object**—ES_NODES

Field	Data Type	Reference	Remarks
DISPLAYED_TE	STRING 32768		
XT			
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID

Field	Data Type	Reference	Remarks
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
LINK_ID	INTEGER	ES_NODES::id	
NODE_ID	INTEGER		
NODE_SHORT_DESC	STRING 150		
NODE_TYPE	INTEGER		
PARENT_NODE_ID	INTEGER		
QUERY_RESP_NUMBER	INTEGER		
QUERY_RESP_TYPE	STRING 50		
RESLINKID1	INTEGER		
RESLINKID2	INTEGER		
RESLINKID3	INTEGER		
RESLINKID4	INTEGER		
RESLINKID5	INTEGER		
RESLINKID6	INTEGER		
RESLINKID7	INTEGER		
RESPONSE1	STRING 100		
RESPONSE2	STRING 100		
RESPONSE3	STRING 100		
RESPONSE4	STRING 100		
RESPONSE5	STRING 100		
RESPONSE6	STRING 100		
RESPONSE7	STRING 100		
ROOT_ID	INTEGER	ES_NODES::id	
TREE_ID	INTEGER	SKELETONS::id	

ES_RESPONSES Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—ES_RESPONSES
- **Object**—ES_RESPONSES

Field	Data Type	Reference	Remarks
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
PARENT_NODE_ID	INTEGER	ES_NODES::id	
RESPONSE_LIN_K_ID	INTEGER	ES_NODES::id	
RESPONSE_LIN_K_ORDER	INTEGER		
RESPONSE_LIN_K_TEXT	STRING 100		

ES_SESSIONS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—ES_SESSIONS
- **Object**—ES_SESSIONS

Field	Data Type	Reference	Remarks
COMMENT_TEXT	STRING 50		
EVALUATION	INTEGER		
EXTERNAL_ID	STRING 50		
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
PATH_IDS	STRING 50		

Field	Data Type	Reference	Remarks
PATH_QAS	STRING 32768		
SESSION_ID	INTEGER		
TREE_ID	INTEGER	ES_NODES::id	

Event_Delay Table

This table lists the times that events were delayed.

- **SQL Name**—evt_dly
- **Object**—evtdly

Field	Data Type	Reference	Remarks
description	STRING 80		User Description of delay
act_delay	DURATION		Actual Duration of delay
cancel_time	LOCAL_TIME		Time delay was cancelled
create_time	LOCAL_TIME		Time delay was created
delay_type	INTEGER	Event_Delay_Type ::enum	enum Event_Delay_Type
eff_delay	DURATION		Effective Duration of delay
group_name	STRING 30		Group Name of attevt (currently only SLA)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
obj_id	STRING 30 NOT_NULL S_KEY		the persid of the object
persid	STRING 30		Persistent ID (SystemObjectName:id)
start_time	LOCAL_TIME		Time delay was started
start_userid	UUID	ca_contact::uuid	user that createdstarted delay

Event_Delay_Type Table

Field	Data Type	Reference	Remarks
status_flag	INTEGER		A flag for indicating an event delay status.
stop_time	LOCAL_TIME		Time delay was stopped
stop_userid	UUID	ca_contact::uuid	user that stopped/cancelled delay
support_lev	STRING 30	Service_Desc::code	service type in effect

Event_Delay_Type Table

User-defined code for the type of event delay.

- **SQL Name**—evtdlytp
- **Object**—evtdlytp

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL	Active_Boolean_Table::enum	Deleted flag (0=active 1=inactive/marketed as deleted)
enum	INTEGER NOT_NULL		Enumerated value for this entry - specifies ordering in lists and relative values
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
nx_desc	STRING 40		
sym	STRING 12 UNIQUE NOT_NULL S_KEY		

event_log Table

USP event log.

- **SQL Name**—event_log
- **Object**—event_log

Field	Data Type	Reference	Remarks
event	INTEGER	event_type::id	Event type
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
kd_id	INTEGER	SKELETONS::id	Assoc knowledge doc
log_time	LOCAL_TIME		Server log date
millitime	INTEGER		Log date millisec
numdata1	INTEGER		Smag number
numdata2	INTEGER		Smag number
sd_obj_id	INTEGER		Assoc SD id
sd_obj_type	STRING 30		Assoc SD object
session	INTEGER	session_log::id	Session with event
textdata1	STRING 500		Smag string
textdata2	STRING 500		Smag string

event_type Table

Event type.

- **SQL Name**—event_type
- **Object**—event_type

Field	Data Type	Reference	Remarks
description	STRING 500		Description
code	STRING 12 UNIQUE NOT_NULL S_KEY		Noneditable string enum
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID

Field	Data Type	Reference	Remarks
last_mod_by	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
persid	STRING 30		Persistent ID (SystemObjectName:id)

Events Table

Tables for Event stuff. Events attached to objects.

- **SQL Name**—evt
- **Object**—evt

Field	Data Type	Reference	Remarks
description	STRING 80		event description
condition	STRING 30	Spell_Macro: :persid	macro function for condition
del	INTEGER NOT_NULL	Active_Boole an_Table::e num	Deleted flag (0=active 1=inactive/marketed as deleted)
delay_time	DURATION NOT_NULL		time until condition check
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
modulo_time	DURATION NOT_NULL		time increment to adjust
obj_type	STRING 30		kind of object for this evt
on_done_flag	INTEGER NOT_NULL		This sets the <i>fire_time</i> time directly on the attached event to indicate when the event is done.

Field	Data Type	Reference	Remarks
persid	STRING 30		Persistent ID (SystemObjectName:id)
sym	STRING 30 NOT_NULL unique S_KEY		event name
urgency	INTEGER		
user_settime	INTEGER NOT_NULL		The user_settime allows the user (or system) to override the fire time (delay time) that is defined for an event. For example, when users add an event to a Service Type, they can redefine the fire time for the event only if the user_settime flag is set. Otherwise, the fire time defined in the event is used.
user_smag	STRING 200		User smag field
violate_on_f	INTEGER else		
violate_on_tr	INTEGER ue		
work_shift	STRING 30	Bop_Workshift::persid	

ext_appl Table

External application.

- **SQL Name**—ext_appl

Field	Data Type	Reference	Remarks
description	STRING 500		Appl description
code	STRING 12 UNIQUE NOT_NULL S_KEY		Noneditable string enum
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID

Field	Data Type	Reference	Remarks
last_mod_by	UUID		Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
persid	STRING 30		Persistent ID (SystemObjectName:id)
sym	STRING 30 NOT_NULL S_KEY		Application name

External_Entity_Map Table

NOT FOR CLIENT USE. Maps an external entity to an internal object where the external entity is uniquely defined in its own namespace by the xentity_id. The namespace is uniquely defined for our use by the xschema_code and xschema_ver. The semantics of the xentity_id, and parameters (*_rsrvd fields) depends upon the namespace.

- **SQL Name**—xent_map
- **Object**—ext_entity_map

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
int1_rsrvd	INTEGER		reserved for CA - do not use
int2_rsrvd	INTEGER		reserved for CA - do not use
int3_rsrvd	INTEGER		reserved for CA - do not use
int4_rsrvd	INTEGER		reserved for CA - do not use
int5_rsrvd	INTEGER		reserved for CA - do not use
int6_rsrvd	INTEGER		reserved for CA - do not use
lstr1_rsrvd	STRING 255		reserved for CA - do not use
lstr2_rsrvd	STRING 255		reserved for CA - do not use
ob_persid	STRING 30		the "mapped to" object
ob_type	STRING 30		the "mapped to" object type

Field	Data Type	Reference	Remarks
persid	STRING 30		Persistent ID (SystemObjectName: id)
str1_rsrved	STRING 80		reserved for CA - do not use
str2_rsrved	STRING 80		reserved for CA - do not use
xentity_id	STRING 180 NOT_NULL		uniq. external entity reference
xschema_cod e	STRING 12 NOT_NULL		internal code for entity's namespace
xschema_ver	INTEGER NOT_NULL		internal ver for entity's namespace

Form_Group Table

Listing of defined form groups.

- **SQL Name**—fmgrp
- **Object**—fmgrp

Field	Data Type	Reference	Remarks
del	INTEGER	Active_ Boolean_ Table:::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
description	nvarchar(100)		Specifies the textual description of this form group.
id	INTEGER		Primary key of this table, it is a unique, numeric ID.
last_mod _by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod _dt	INTEGER		Indicates the timestamp of when this record was last modified.
sym	nvarchar(30)		Specifies the symbolic value for this Form_Group.

Global_Change_Extension Table

Local copy of data that will be pushed to master's Global_Change_Queue table.

- **SQL Name**—g_chg_ext
- **Object**—g_chg_ext

Field	Data Type	Reference	Remarks
active_flag	INTEGER NOT_NULL	Active_Reverse _Boolean::enu m	(1=Active, 0=not active)
affected_contact	UUID NOT_NULL	ca_contact::uui d	Affected End User of CO
assignee	UUID	ca_contact::uui d	Assignee of CO
category	STRING 30		Category symbol of CO
chg_ref_num	STRING 30 NOT_NULL		Change ref num
close_date	LOCAL_TIME		Close Date of CO
global_queue_id	INTEGER	Global_Queue_ Names::id	Pointer to global queue
group_id	UUID		Group assigned to CO
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
impact	INTEGER	Impact::enum	Impact of Change order
last_mod_dt	LOCAL_TIME NOT_NULL		Timestamp of when this record was last modified
open_date	LOCAL_TIME NOT_NULL		Open Date of CO
priority	INTEGER NOT_NULL	Priority::enum	Priority of CO
remote_id	INTEGER NOT_NULL S_KEY		Regional request id

Field	Data Type	Reference	Remarks
requestor	UUID NOT_NULL	ca_contact::uuid	Requestor of CO
status	STRING 30 NOT_NULL	Change_Status::code	Status symbol of CO
summary	STRING 240		Summary of CO

Global_Change_Queue Table

This table is an accumulation of all of the regional Global_Change_Extension tables.

- **SQL Name**—g_chg_queue
- **Object**—g_chg_queue

Field	Data Type	Reference	Remarks
active_flag	INTEGER NOT_NULL	Active_Reverse_Boolean::enum	(1=Active, 0=not active)
affected_contact	UUID NOT_NULL		Affected End User of CO
assignee	UUID		Assignee of CO
category	STRING 30		Category symbol of CO
chg_ref_num	STRING 30 NOT_NULL		Change ref num
close_date	LOCAL_TIME		Close Date of CO
global_queue_id	INTEGER	Global_Queue_Names::id	Pointer to global queue
group_id	UUID		Group assigned to CO
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
impact	INTEGER	Impact::enum	Impact of Change order
last_mod_dt	LOCAL_TIME NOT_NULL		Timestamp of when this record was last modified

Global_Contact Table

Field	Data Type	Reference	Remarks
open_date	LOCAL_TIME NOT_NULL		Open Date of CO
priority	INTEGER NOT_NULL	Priority::enum	Priority of CO
remote_id	INTEGER NOT_NULL S_KEY		Regional change id
remote_sys_id	INTEGER NOT_NULL S_KEY	Global_Servers: :remote_sys_id	Regional system id
requestor	UUID NOT_NULL		Requestor of CO
status	STRING 30 NOT_NULL	Change_Status: :code	Status symbol of CO
summary	STRING 240		Summary of CO

Global_Contact Table

Contain all contacts across all systems participating in the Global Service Desk.

- **SQL Name**—g_contact
- **Object**—g_cnt

Field	Data Type	Reference	Remarks
contact_num	STRING 30		Contact num
del	INTEGER NOT_NULL	Active_Boolean _Table::enum	Deleted flag (0=active 1=inactive/mark as deleted)
email_address	STRING 120		email address
first_name	STRING 100		Contact's first name
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
last_name	STRING 100		Contact's last name
loc_id	UUID		Location id

Field	Data Type	Reference	Remarks
middle_name	STRING 100		Contact's middle name
org_id	UUID		Organization id
pri_phone_number	STRING 32		phone number
remote_id	UUID NOT_NULL S_KEY		Regional contact id
remote_system_id	INTEGER NOT_NULL S_KEY	Global_Servers::remote_system_id	Regional system id
userid	STRING 100		Contact's userid

Global_Issue_Extension Table

Local copy of data that will be pushed to master's Global_Issue_Queue table.

- **SQL Name**—g_iss_ext
- **Object**—g_iss_ext

Field	Data Type	Reference	Remarks
active_flag	INTEGER NOT_NULL	Active_Reverse_Boolean::enum	(1=Active, 0=not active)
assignee	UUID	ca_contact::uuid	Assignee of Issue
category	STRING 30		Category symbol of Issue
close_date	LOCAL_TIME		Close Date of Issue
global_queue_id	INTEGER	Global_Queue_Names::id	Pointer to global queue
group_id	UUID		Group assigned to Issue
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
impact	INTEGER	Impact::enum	Impact of Issue
last_modified	LOCAL_TIME NOT_NULL		Timestamp of when this record was last modified
open_date	LOCAL_TIME NOT_NULL		Open Date of Issue

Global_Issue_Queue Table

Field	Data Type	Reference	Remarks
priority	INTEGER NOT_NULL	Priority::enum	Priority of Issue
product	INTEGER	Product::id	Product of Issue
ref_num	STRING 30 NOT_NULL		Issue ref num
remote_id	INTEGER NOT_NULL S_KEY		Regional request id
requestor	UUID NOT_NULL	ca_contact::uuid	Affected End User of CO
status	STRING 30 NOT_NULL	Issue_Status::co_de	Status symbol of Issue
summary	STRING 240		Summary of Issue

Global_Issue_Queue Table

This table is an accumulation of all of the regional Global_Issue_Extension tables.

- **SQL Name**—g_iss_queue
- **Object**—g_iss_queue

Field	Data Type	Reference	Remarks
active_flag	INTEGER NOT_NULL	Active_Reverse_Boolean::enum	(1=Active, 0=not active)
assignee	UUID		Assignee of Issue
category	STRING 30		Category symbol of Issue
close_date	LOCAL_TIME		Close Date of Issue
global_queue_id	INTEGER	Global_Queue_Names::id	Pointer to global queue
group_id	UUID		Group assigned to Issue
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
impact	INTEGER	Impact::enum	Impact of Issue
last_mod_dt	LOCAL_TIME NOT_NULL		Timestamp of when this record was last modified

Field	Data Type	Reference	Remarks
open_date	LOCAL_TIME NOT_NULL		Open Date of Issue
priority	INTEGER NOT_NULL	Priority::enum	Priority of Issue
product	INTEGER		Product of Issue
ref_num	STRING 30 NOT_NULL		Issue ref num
remote_id	INTEGER NOT_NULL S_KEY		Regional Issue id
remote_sys_id	INTEGER NOT_NULL S_KEY	Global_Servers:: remote_sys_id	Regional system id
requestor	UUID NOT_NULL		Affected User of Issue
status	STRING 30 NOT_NULL	Issue_Status::code	Status symbol of Issue
summary	STRING 240		Summary of Issue

Global_Location Table

Contains all Location names for all regions.

- **SQL Name**—g_loc
- **Object**—g_loc

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL	Active_Boolean_Table::enum	Deleted flag (0=active 1=inactive/marketed as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
loc_name	STRING 100		Location Name
remote_id	UUID NOT_NULL S_KEY		Regional location id

[Global_Organization Table](#)

Field	Data Type	Reference	Remarks
remote_sys_id	INTEGER NOT_NULL S_KEY	Global_Servers: :remote_sys_id	Regional system id

Global_Organization Table

Contains all Organizations names for all regions.

- **SQL Name**—g_org
- **Object**—g_org

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL	Active_Boolean _Table::enum	Deleted flag (0=active 1=inactive/mark as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
org_name	STRING 100		Organization Name
remote_id	UUID NOT_NULL S_KEY		Regional organization id
remote_sys_id	INTEGER NOT_NULL S_KEY	Global_Servers: :remote_sys_id	Regional system id

Global_Product Table

Contains all Product names for all regions. Used by Global_Issue_Queue.

- **SQL Name**—g_product
- **Object**—g_prod

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL	Active_Boolean_ Table::enum	Deleted flag (0=active 1=inactive/marketed as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
remote_id	INTEGER NOT_NULL S_KEY		Regional product id
remote_sys_id	INTEGER NOT_NULL S_KEY	Global_Servers:: remote_sys_id	Regional system id
sym	STRING 60		Symbol name of product

Global_Queue_Names Table

List of queues that are considered global.

- **SQL Name**—g_queue_names
- **Object**—g_qname

Field	Data Type	Reference	Remarks
description	STRING 100		comments field
del	INTEGER NOT_NULL	Active_Boolean_ Table::enum	Deleted flag (0=active 1=inactive/marketed as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID

Global_Request_Extension Table

Field	Data Type	Reference	Remarks
last_mod_by	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
sym	STRING 30 UNIQUE NOT_NULL		Descriptive name

Global_Request_Extension Table

Local copy of data that will be pushed to master's Global_Request_Queue table.

- **SQL Name**—g_req_ext
- **Object**—g_cr_ext

Field	Data Type	Reference	Remarks
active_flag	INTEGER NOT_NULL	Active_Reverse_Boolean::enum	(1=Active, 0=not active)
assignee	UUID	ca_contact::uuid	Assignee of Request
category	STRING 30		Category symbol of Request
close_date	LOCAL_TIME		Close Date of Request
customer	UUID NOT_NULL	ca_contact::uuid	Affected User of Request
global_queue_id	INTEGER	Global_Queue_Name::id	Pointer to global queue
group_id	UUID		Group assigned to Request
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
impact	INTEGER	Impact::enum	Impact of Request
last_mod_dt	LOCAL_TIME NOT_NULL		Timestamp of when this record was last modified
open_date	LOCAL_TIME NOT_NULL		Open Date of Request

Field	Data Type	Reference	Remarks
priority	INTEGER NOT_NULL	Priority::enum	Priority of Request
ref_num	STRING 30 NOT_NULL		Request ref num
remote_id	INTEGER NOT_NULL S_KEY		Regional request id
status	STRING 30 NOT_NULL	Cr_Status::code	Status symbol of Request
summary	STRING 240		Summary of Request
type	STRING 10	crt code	Itil record type

Global_Request_Queue Table

This table is an accumulation of all of the regional Global_Request_Extension tables.

- **SQL Name**—g_req_queue
- **Object**—g_cr_queue

Field	Data Type	Reference	Remarks
active_flag	INTEGER NOT_NULL	Active_Reverse_ Boolean::enum	(1=Active, 0=not active)
assignee	UUID		Assignee of Request
category	STRING 30		Category symbol of Request
close_date	LOCAL_TIME		Close Date of Request
customer	UUID NOT_NULL		Affected User of Request
global_queue_id	INTEGER	Global_Queue_Names::id	Pointer to global queue
group_id	UUID		Group assigned to Request
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
impact	INTEGER	Impact::enum	Impact of Request
last_mod_dt	LOCAL_TIME NOT_NULL		Timestamp of when this record was last modified

Global_Servers Table

Field	Data Type	Reference	Remarks
open_date	LOCAL_TIME NOT_NULL		Open Date of Request
priority	INTEGER NOT_NULL	Priority::enum	Priority of Request
ref_num	STRING 30 NOT_NULL		Request ref num
remote_id	INTEGER NOT_NULL S_KEY		Regional request id
remote_sys_id	INTEGER NOT_NULL S_KEY	Global_Servers:::remote_sys_id	Regional system id
status	STRING 30 NOT_NULL	Cr_Status::code	Status symbol of Request
summary	STRING 240		Summary of Request
type	STRING 10	crt code	Itil record type

Global_Servers Table

Maintains a list of Service Desk installations that comprise the Global Service Desk.

- **SQL Name**—g_srvr
- **Object**—g_svrs

Field	Data Type	Reference	Remarks
description	STRING 100		comments field
chg_prefix	STRING 5		prefix for changes for a region
cr_prefix	STRING 5		prefix for requests for a region
del	INTEGER NOT_NULL	Active_Bool ean_Table:::enum	Deleted flag (0=active 1=inactive/mark as deleted)
global_name	STRING 30 UNIQUE NOT_NULL		Value of NX_GLOBAL_NAME on region

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
is_master	INTEGER	Boolean_Ta ble::enum	Is this server defined as master
iss_prefix	STRING 5		prefix for issues for a region
last_mod_by	UUID	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIM E		Timestamp of when this record was last modified
remote_sys_ id	INTEGER UNIQUE NOT_NULL		Regional system id
slump_addr	STRING 30		Specifies the Service Desk primary server hostname or IP address. A slump_addr is defined for each global region definition.
sym	STRING 30 UNIQUE NOT_NULL		Descriptive name
web_protocol	STRING 10		master region web access protocol (http https)
web_server	STRING 30		web server name
web_server_ port	STRING 10		web server port(can be blank)
web_url	STRING 100		rest of URL to pdmweb.exe of remote system (or webdirector)

Global_Table_Map Table

Contains list of table maps (local to master) used in global data transport.

- **SQL Name**—g_tbl_map
- **Object**—g_tblmap

Field	Data Type	Reference	Remarks
description	STRING 100		Comments field
del	INTEGER NOT_NULL	Active_Boolean_ Table::enum	Deleted flag (0=active 1=inactive/marketed as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_b_y	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
map_definition	STRING 64 NOT_NULL		Map Definition name in xml file
sym	STRING 30 UNIQUE NOT_NULL		Name of this table map

Global_Table_Rule Table

Contains list of extract rules for global data transport.

- **SQL Name**—g_tbl_rule
- **Object**—g_tblrule

Field	Data Type	Reference	Remarks
description	STRING 100		Comments field
addl_query	STRING 240		Additional query

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL	Active_Boolean_Table::enum	Deleted flag (0=active 1=inactive/mark as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_b_y	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_d_t	LOCAL_TIME		Timestamp of when this record was last modified
last_sync_d_t	LOCAL_TIME		What was the last sync date
reoccur_interv	DURATION		How often to re-query
sched	STRING 30	Bop_Workshift::p_ersid	Valid run schedule
sym	STRING 30 UNIQUE NOT_NULL S_KEY		Name for rule
table_map	INTEGER	Global_Table_Map_p::id	Table map

Group_Loc Table

Used to build the list of locations a group can service.

- **SQL Name**—grp_loc
- **Object**—grp_loc

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		left hand attribute name

Field	Data Type	Reference	Remarks
l_persid	STRING 60		left hand key
l_sql	INTEGER		left hand sort order
r_attr	STRING 30		right hand attribute name
r_persid	STRING 60		right hand key
r_sql	INTEGER		right hand sort order

Group_Member Table

Group Members.

- **SQL Name**—grpmem
- **Object**—grpmem

Field	Data Type	Reference	Remarks
group_id	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is the group.
id	INTEGER		Specifies the unique (to the table) numeric ID.
manager_flag	INTEGER		Specifies the Manager flag, as follows: 0=No 1=Group manager
member	byte(16)	ca_contact ::uuid	Foreign key to the contact_uuid field of the ca_contact table, this is the Member.
notify_flag	INTEGER		Specifies the Notify flag, as follows: 0>No notification 1=Notify

Impact Table

Impact is a measure of the significance of an event by the user. It is used on Incident Reports.

- **SQL Name**—impact
- **Object**—imp

Field	Data Type	Reference	Remarks
del	INTEGER	Active_ Boolean_ Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
enum	INTEGER		Primary key for this table.
id	INTEGER		Specifies the unique (to the table) numeric ID.
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
nx_desc	nvarchar(40)		Specifies the textual description of this impact.
sym	nvarchar(12)		Identifies the symbolic value for this impact.
value	INTEGER		Shows the numeric representation of this impact.

INDEX_DOC_LINKS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—INDEX_DOC_LINKS
- **Object**—INDEX_DOC_LINKS

Field	Data Type	Reference	Remarks
DOC_ID	INTEGER	SKELETONS::id	

Field	Data Type	Reference	Remarks
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
INDEX_ID	INTEGER	O_INDEXES::id	
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
RELATIONAL_ID	STRING 255		

Interface Table

Interface definitions used for creating requests and change orders.

- **SQL Name**—interface
- **Object**—intfc

Field	Data Type	Reference	Remarks
code	nvarchar(10)		Specifies the code value for this interface.
del	INTEGER	Active_ Boolean_ Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
id	INTEGER		Primary key of this table, it is a unique numeric ID.
last_mod _by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
nx_desc	nvarchar (240)		The desc value for this interface.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sym	nvarchar(30)		Identifies the symbolic value for this interface.

ISS_Template Table

Table to manage Issue templates. There is one entry for each Issue that is a template.

- **SQL Name**—iss_template
- **Object**—iss_tpl

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
description	nvarchar(1000)		Specifies the textual description of the template.
id	INTEGER		Specifies the unique (to the table) numeric ID.
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
quick_tmpl_type	INTEGER	Quick_Template_Types ::enum	Specifies the quick template type, as follows: 1=Quick tmpl 2=Quick tmpl+review
template	nvarchar(30)	issue_persistent_id	Persistent ID (SystemObjectName:id).
template_class	nvarchar(12)		This allow subclassing of the templates.
template_name	nvarchar(30)		Identifies the unique name of the template.

Isiscat_Group Table

Used to build a list of Groups that can service the Category.

- **SQL Name**—icat_grp
- **Object**—isiscat_grp

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		left hand attribute name
l_persid	STRING 60		left hand key
l_sql	INTEGER		left hand sort order
r_attr	STRING 30		right hand attribute name
r_persid	STRING 60		right hand key
r_sql	INTEGER		right hand sort order

Isiscat_Loc Table

Used to build a list of Service Locations valid for the Category.

- **SQL Name**—icat_loc
- **Object**—isiscat_loc

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		left hand attribute name
l_persid	STRING 60		left hand key
l_sql	INTEGER		left hand sort order
r_attr	STRING 30		right hand attribute name
r_persid	STRING 60		right hand key
r_sql	INTEGER		right hand sort order

Isscat_Workshift Table

Used to build the list of valid service workshifts for the Category.

- **SQL Name**—icat_wrkshft
- **Object**—isscat_workshift

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		left hand attribute name
l_persid	STRING 60		left hand key
l_sql	INTEGER		left hand sort order
r_attr	STRING 30		right hand attribute name
r_persid	STRING 60		right hand key
r_sql	INTEGER		right hand sort order

Issue Table

Request management's analyst recording of an external user's ticket.

- **SQL Name**—issue
- **Object**—iss

Field	Data Type	Reference	Remarks
actions	nvarchar (750)		Identifies the actions value for this Issue.
active_flag	INTEGER		Flag representing whether this record is active or inactive: 0=Inactive 1=Active
actual_comp_date	INTEGER		Specifies the actual completion date timestamp.
actual_cost	INTEGER		Identifies the actual cost of this Issue.

Field	Data Type	Reference	Remarks
actual_total_time	INTEGER		Specifies the sum of actual task time.
affected_contact	byte(16)	ca_contact ::uuid	Foreign key to the contact_uuid field of the ca_contact table, this specifies the Affected End User.
assignee	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this identifies the Assignee.
backout_plan	nvarchar(240)		Identifies the backout plan value for this Issue.
call_back_date	INTEGER		Identifies the call back timestamp for this Issue.
call_back_flag	INTEGER		Specifies the call back flag value for this Issue.
category	nvarchar(12)	Issue Category::id	Foreign key to the code field of the isscat table, this identifies the Category.
cawf_procid	nvarchar(40)		Identifies the CA Workflow process id.
close_date	INTEGER		Shows the timestamp of when the Issue was closed.
created_via	INTEGER	Interface::id	Specifies the unique (to the table) numeric ID.
description	nvarchar(4000)		Provides a textual description of this Issue.
effort	nvarchar(240)		Identifies the effort value for this Issue.
est_comp_date	INTEGER		Shows the estimated completion date for this Issue.
est_cost	INTEGER		Specifies the estimated cost value for this Issue.
est_total_time	INTEGER		Identifies the sum of estimated task time.

Field	Data Type	Reference	Remarks
flag1	INTEGER		This is the flag1 value for this Issue.
flag2	INTEGER		This is the flag2 value for this Issue.
flag3	INTEGER		This is the flag3 value for this Issue.
flag4	INTEGER		This is the flag4 value for this Issue.
flag5	INTEGER		This is the flag5 value for this Issue.
flag6	INTEGER		This is the flag6 value for this Issue.
group_id	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is the Group ID.
id	INTEGER		Specifies the unique (to the table) numeric ID.
impact	INTEGER	Impact::enum	Foreign key to the enum field of the impact table, this defines the impact.
justification	nvarchar(240)		Identifies the justification value for this Issue.
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
log_agent	byte(16)	ca_contact::uuid	Foreign key to the contact_uuid field of the ca_contact table, this identifies who the change request was reported by.
macro_predict_viol	INTEGER		Identifies that it is likely to violate its sla (boolean) for action macros to predict sla violations.

Field	Data Type	Reference	Remarks
need_by	INTEGER		Shows the Need By Date timestamp for this Issue..
open_date	INTEGER		Shows the timestamp of when this Issue was opened.
organization	byte(16)	ca_organization::organization_uuid	Foreign key to the id field of the ca_organization table, this identifies the Organization.
parent	nvarchar(30)	issue persistent_id	Foreign key to the persistent_id field of the iss table, this identifies the Parent.
persid	nvarchar(30)		Specifies the Persistent ID (SystemObjectName:id).
person_contacting	INTEGER	Person_Contacting::id	Foreign key to the id field of the person table, this identifies the person who made the contact.
predicted_sla_viol	INTEGER		Flag that indicates the following: 1=Has been predicted by neugents.
priority	INTEGER	Priority::enum	Foreign key to the enum field of the pri table, this identifies the Priority.
product	INTEGER	Product::id	Foreign key to the id field of the product table, this identifies the Product.
ref_num	nvarchar(30)		Identifies the reference number.
reporting_method	INTEGER	Reporting_Method::id	Foreign key to the id field of the repmeth table, this identifies how this Issue was reported.
requestor	byte(16)	ca_contact::uuid	Foreign key to the contact_uuid field of the ca_contact table, this identifies the Requestor.

Field	Data Type	Reference	Remarks
resolve_date	INTEGER		Indicates the timestamp of when the Change Order was resolved.
rootcause	INTEGER	Rootcause::id	Foreign key to the id field of the rootcause table, this identifies the Root Cause.
service_date	INTEGER		Specifies the service date value for this Issue.
service_num	nvarchar(30)		Specifies the service num value for this Issue.
slaViolation	INTEGER		Flag indicating the following: 1=Issue has violated its sla.
start_date	INTEGER		Specifies the timestamp of when processing started.
status	nvarchar(12)	Issue Status::code	Foreign key to the code field of the issstat table, this identifies the Status.
string1	nvarchar(40)		This is a user-defined string field.
string2	nvarchar(40)		This is a user-defined string field.
string3	nvarchar(40)		This is a user-defined string field.
string4	nvarchar(40)		This is a user-defined string field.
string5	nvarchar(40)		This is a user-defined string field.
string6	nvarchar(40)		This is a user-defined string field.
summary	nvarchar(240)		Identifies the Issue summary text.
support_lev	nvarchar(30)	Service_Desc ::code	Foreign key to the code field of the srv_desc table, this identifies the Classic Service Type.

Issue_Act_Log Table

Field	Data Type	Reference	Remarks
template_name	nvarchar(30)	iss_template::template_name	Foreign key to the template_name field of the iss_tpl table, this identifies the Template name.
type_of_contact	INTEGER	Type_of_Contact::id	Foreign key to the id field of the toc table, this identifies the Type of Contact.
user1	nvarchar(100)		This is a user-defined string field.
user2	nvarchar(100)		This is a user-defined string field.
user3	nvarchar(100)		This is a user-defined string field.

Issue_Act_Log Table

History of activities associated with an issue. Types of activities are listed in the Act_Type table.

- **SQL Name**—issalg
- **Object**—issalg

Field	Data Type	Reference	Remarks
action_desc	nvarchar(4000)		Specifies the text description of the activity log entry.
analyst	byte(16)	ca_contact::uuid	Foreign key to the contact_uuid field of the ca_contact table, this represents the Analyst who created this activity log.
description	nvarchar(4000)		Specifies the text description of the activity log.
id	INTEGER		Primary key of this table, this is a unique numeric ID.

Field	Data Type	Reference	Remarks
internal	INTEGER		Designates the log as an Internal log.
issue_id	nvarchar(30)	issue persistent _id	Foreign key to the persistent_id field of the iss table, this is the Issue for this activity.
knowledge_session	nvarchar(80)		Specifies the reference to the Knowledge Tool session.
knowledge_tool	nvarchar(12)		Identifies the Knowledge Tool used for this activity.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
persid	nvarchar(30)		Specifies the Persistent ID (SystemObjectName: id).
system_time	INTEGER		Represents the date and time of the record creation.
time_spent	INTEGER		Identifies the time spent on the activity by the user.
time_stamp	INTEGER		Specifies the time spent on the activity by the user.
type	nvarchar(12)	Act_Type ::code	(Not Used) This is an acknowledgement that is also a non-editable string enum.

Issue_Category Table

Issue categories. Can be hierarchical.

- **SQL Name**—isscat
- **Object**—isscat

Field	Data Type	Reference	Remarks
assignee	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is also the Assignee.
auto_assign	INTEGER		Flag that enables auto assignment.
cawf_defid	nvarchar(40)		Identifies the CA Workflow definition id.
children_ok	INTEGER		Specifies the handling of the issue category: 0=Children not allowed 1=Children are allowed
code	nvarchar(30)		Primary key of this table.
del	INTEGER	Active_Boolean _Table::enum	Delete flag that indicates the following: 0=Active 1=Inactive/mark as deleted
description	nvarchar(500)		Identifies the textual description of this issue category.
group_id	byte(16)		Foreign key to the contact_uuid field of the ca_contact table. This is the Group.
id	INTEGER		Unique (to the table) numeric ID.
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
organization	byte(16)	ca_organization ::organization _uuid	Foreign key to the id field of the ca_organization table, this is the Organization.

Field	Data Type	Reference	Remarks
owning_contract	INTEGER	Service_Contract::id	Foreign key to the id field of the svc_contract table. This is the Service Contract.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id)
schedule	INTEGER		Deprecated.
survey	INTEGER	Survey_Template ::id	Foreign key of the id field of the survey_tpl table, this is the Survey.
sym	nvarchar(30)		Identifies the Issue Category symbolic description.

Issue_Property Table

Property value pairs for an object.

- **SQL Name**—issprp
- **Object**—iss_prp

Field	Data Type	Reference	Remarks
description	nvarchar(240)		Specifies the textual description of this property.
id	INTEGER		Primary key of this table, it is also a unique, numeric ID.
label	nvarchar(80)		Specifies the label value for this Issue property.
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
owning_iss	nvarchar(30)	issue_persistent_id	Foreign key to the persistent_id field of the iss table, this is also the Issue.

Field	Data Type	Reference	Remarks
required	INTEGER	Boolean _Table::: enum	Identifies the Required flag, as follows: 0=Not required 1=Required
sample	nvarchar(240)		The sample value for this property.
sequence	INTEGER		Used to order the properties for an Issue.
value	nvarchar(240)		Identifies the value of the property.

Issue_Status Table

Lists the states of the Issue, which you can also add. This table allows you to control whether the issue is active or inactive when it is changed to this status. Possible status include: open, approval in process, implementation in progress, verification in progress, cancelled, suspended, and closed.

- **SQL Name**—issstat
- **Object**—issstat

Field	Data Type	Reference	Remarks
active	INTEGER		Flag that indicates the following: 0=Inactive 1=Active
code	nvarchar(12)		Primary key of this table.
del	INTEGER		Delete flag that indicates the following: 0=Active 1=Inactive/mark as deleted
description	nvarchar(500)		Identifies the textual description of this status.
hold	INTEGER		Flag that specifies the following: 0=Start events 1=Stop events
id	INTEGER		Unique (to the table) numeric ID.
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.

Field	Data Type	Reference	Remarks
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName: id).
resolved	INTEGER		Flag that indicates the following: 0=Not yet resolved 1=Resolved
sym	nvarchar(30)		Identifies the symbolic name of the Issue status.

Issue_Workflow_Task Table

Object issue workflow tasks. A task is in this table because it can be added from an issue, which means there is no task template from which to obtain the task.

- **SQL Name**—isswf
- **Object**—iss_wf

Field	Data Type	Reference	Remarks
actual_duration	INTEGER		Specifies the time spent on the task by the user.
asset	byte(16)	ca_owned_resource::uuid	Foreign key to the id field of the ca_owned_resource table, this is the Asset.
assignee	byte(16)	ca_contact::uuid	Foreign key to the contact_uuid field of the ca_contact table, this is the Assignee.
completion_date	INTEGER		Specifies the timestamp of when this task was completed.
cost	INTEGER		Specifies the cost value for this Issue Workflow task.

Issue_Workflow_Task Table

Field	Data Type	Reference	Remarks
creator	byte(16)	ca_contact ::uuid	Foreign key to the contact_uuid field of the ca_contact table, this identifies the creator of the Issue Workflow task.
date_created	INTEGER		Specifies the date and time that the record was created.
del	INTEGER		Specifies the Deleted flag as follows: 0=Active 1=Inactive/marketed as deleted
description	nvarchar(1000)		Specifies workflow task notes.
done_by	byte(16)	ca_contact ::uuid	Foreign key to the contact_uuid field of the ca_contact table, this provides the Completed by information.
est_comp_date	INTEGER		The estimated completion date value for this Issue Workflow task.
est_cost	INTEGER		The estimated cost value for this Issue Workflow task.
est_duration	INTEGER		The estimated duration value for this Issue Workflow task.
group_id	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is the Group.
group_task	INTEGER		Defines the Group task flag, as follows: 0=No 1=Yes
id	INTEGER		Primary key of this table.

Field	Data Type	Reference	Remarks
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified
object_id	nvarchar(30)		Persistent ID (SystemObjectName:id)
object_type	nvarchar(30)		Indicates the short name of the object to which this workflow task belongs.
persid	nvarchar(30)		Specifies the Persistent ID (SystemObjectName:id).
sequence	INTEGER		Specifies the sequence value for this Issue Workflow task.
start_date	INTEGER		Specifies the timestamp of when the status went to pending.
status	nvarchar(12)	Task_Status ::code	Foreign key to the code field of the tskstat table, this is the Task status.
support_lev	nvarchar(30)	Service_Desc::code	Foreign key to the code field of the srv_desc table, this identifies the Classic Service Type.
task	nvarchar(12)	Task_Type ::code	Foreign key to the code field of the tskty table, this is the Task template.
wf_template	INTEGER	Workflow_Task_Template ::id	Foreign key to the id field of the wftpl table, this is the Workflow Task template.

KD_ATTMNT Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—KD_ATTMNT
- **Object**—KD_ATTMNT

Field	Data Type	Reference	Remarks
ATTMNT_ID	INTEGER	Attachment::id	
DOC_ID	INTEGER	SKELETONS::id	
ID	INTEGER KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified

kdlinks Table

Each record indicates a link between a KD and request or issues link_type field represented by the following enum value:

1 = KD is a solution to the ticket

2 = Ticket created is based on the document last_mod_by the person that links the document and the ticket

- **SQL Name**—kdlinks
- **Object**—kdlinks

Field	Data Type	Reference	Remarks
cr	STRING 30	Call_Req::persid	
ID	INTEGER KEY		Specifies the Numeric ID that is unique to this table.
iss	STRING 30	issue persistent_id	
kd	INTEGER	SKELETONS::id	
last_mod_by	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.

Field	Data Type	Reference	Remarks
LAST_MOD_DT	LOCAL_TIME		Indicates the Timestamp for when this record was last modified.
link_type	INTEGER		
sd_obj_id	INTEGER		
sd_obj_type	STRING 5		

Key_Control Table

Table of sequence numbers for requests, change orders, tickets, and foreign keys.

- **SQL Name**—kc

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
key_name	STRING 20		
key_value	INTEGER		Indicates the next key available for handout.

Knowledge_Keywords Table

Knowledge Base key words for associating trouble codes and call areas to solutions.

- **SQL Name**—km_kwrd
- **Object**—kwrd

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified

Knowledge_Lrel_Table Table

Field	Data Type	Reference	Remarks
persid	STRING 30		Persistent ID (SystemObjectName:id)
sym	STRING 30 NOT_NULL S_KEY		keyword

Knowledge_Lrel_Table Table

Knowledge base many to many relationships for keywords and call area or trouble code. Do not use this table. Fields Must be kept in sync with Lrel_Table in bop_schema.sch.

- **SQL Name**—km_lrel
- **Object**—kmlrel

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		left hand attribute name
l_persid	STRING 60		left hand key
l_sql	INTEGER		left hand sort order
r_attr	STRING 30		right hand attribute name
r_persid	STRING 60		right hand key
r_sql	INTEGER		right hand sort order

KT_REPORT_CARD Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—KT_REPORT_CARD
- **Object**—KT_REPORT_CARD

Field	Data Type	Reference	Remarks
AVERAGE_EFFECTIVE	INTEGER		NESS_RATING

Field	Data Type	Reference	Remarks
creation_date	INTEGER		The timestamp indicating when this record was created
creation_user	STRING 64		The name of the person who created this record. Should be in form: LastName, FirstName
DOCUMENTS_PUBLISHED	INTEGER		
DOCUMENTS_SUBMITTED	INTEGER		
ID	INTEGER KEY		Unique (to the table) Numeric ID
last_update_date	INTEGER		Timestamp of when this record was last modified
last_update_user	STRING 64		Specifies the contact who last modified this record. Should be in form: LastName, FirstName
ORG_STATISTICS	INTEGER		
PAST_DAYS	INTEGER		
SUBJECT_ID	STRING 40		
Total_votes	INTEGER		
Avg_rating	FLOAT		
User_slv_cnt	INTEGER		
TOTAL_HITS	INTEGER		
TOTAL SOLUTION COUNT	INTEGER		

LONG_TEXTS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—LONG_TEXTS
- **Object**—LONG_TEXTS

Field	Data Type	Reference	Remarks
ACTUAL_TEXT	STRING 32768		
CNT_ORDER	INTEGER		
ID	INTEGER KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
REF_PERSID	STRING 30		

Lrel_Table Table

Must be kept in sync with Lrel_Table_Archive.

- **SQL Name**—lrel
- **Object**—lrel1

Field	Data Type	Reference	Remarks
id	INTEGER		Specifies the unique (to the table) numeric ID.
l_attr	nvarchar(30)		Indicates the left attribute value for this Lrel_Table.
l_persid	nvarchar(30)		Persistent ID (SystemObjectName:id) of the left side of the relationship.
l_sql	INTEGER		Specifies the left sequence value for this Lrel_Table.

Field	Data Type	Reference	Remarks
r_attr	nvarchar(30)		Indicates the right attribute value for this Lrel_Table.
r_persid	byte(16)		Persistent ID (SystemObjectName:id) of the right side of the relationship.
r_sql	INTEGER		Specifies the right sequence value for this Lrel_Table.

Managed_Survey Table

Stores the definition of the managed surveys.

- **SQL Name**—managed_survey
- **Object**—mgs

Field	Data Type	Reference	Remarks
create_date	INTEGER		Identifies when the managed survey was created.
del	INTEGER	Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/mark as deleted
description	nvarchar(400)		Identifies the textual description of this managed survey.
end_date	INTEGER		Specifies the end of survey period.
group_id	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is the Group.
id	INTEGER		Primary key of this table.

Field	Data Type	Reference	Remarks
initial_method	INTEGER	Contact_Method::id	Foreign key to the id field of the ct_mth table, this specifies the Contact Method.
initial_msgbody	nvarchar(1000)		Identifies the message body of the initial notification message.
initial_msgttitle	nvarchar(80)		Identifies the message title of the initial notification message.
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
owner	byte(16)	ca_contact ::uuid	Foreign key to the contact_uuid field of the ca_contact table, this specifies the Owner.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id)
reminder_method	INTEGER	Contact_Method::id	Foreign key to the id field of the ct_mth table, this is the Reminder Contact Method.
reminder_msgbody	nvarchar(1000)		Identifies the reminder message body of the initial notification message.
reminder_msmttitle	nvarchar(80)		Identifies the message title of the initial notification message.
start_date	INTEGER		Specifies the start of the survey period.
status	nvarchar(12)	Mgs_Status::code	Foreign key to the code field of the mgsstat table, this is the Status.

Field	Data Type	Reference	Remarks
sym	nvarchar(12)		Identifies the symbolic name for this Managed Survey.
tplid	INTEGER	Survey_Template ::id	Foreign key to the id field of the survey_tpl table, this is the Survey Template.

Mgs_Act_Log Table

History of activities associated with a managed survey. The types of activities are listed in the Act_Type table.

- **SQL Name**—mgsalg
- **Object**—mgsalg

Field	Data Type	Reference	Remarks
action_desc	nvarchar(700)		Specifies the text description of the activity log entry.
analyst	byte(16)	ca_contact ::uuid	Foreign key to the contact_uuid field of the ca_contact table, this represents the Analyst who created this activity log.
description	nvarchar(1000)		Specifies the text description of the activity log.
id	INTEGER		Primary key of this table, this is a unique numeric ID.
internal	INTEGER		Designates the log as an Internal log.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.

Field	Data Type	Reference	Remarks
mgs_id	INTEGER	Managed_Survey::id	Specifies the unique (to the table) numeric ID.
persid	nvarchar(30)		Specifies the Persistent ID (SystemObjectName:id).
system_time	INTEGER		Represents the date and time of the record creation.
time_spent	INTEGER		Identifies the time spent on the activity by the user.
time_stamp	INTEGER		Specifies the date time spent on the activity by the user.
type	nvarchar(12)	Act_Type::code	Foreign key to the code field of the act_type table, this is the Type.

Mgs_Status Table

List of valid message status definitions.

- **SQL Name**—mgsstat
- **Object**—mgsstat

Field	Data Type	Reference	Remarks
active	INTEGER		Flag that indicates the following: 0=Inactive 1=Active
code	nvarchar(12)		Primary key of this table.
del	INTEGER	Active_Boolean_Table::enum	Delete flag that indicates the following: 0=Active 1=Inactive/mark as deleted
description	nvarchar(500)		Identifies the textual description of this status.

Field	Data Type	Reference	Remarks
hold	INTEGER		Flag that specifies the following: 0=Start events 1=Stop events
id	INTEGER		Unique (to the table) numeric ID.
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sym	nvarchar(30)		Identifies the Managed Survey Status name.

Note_Board Table

Message board (announcements) on the main menu.

- **SQL Name**—cnote
- **Object**—cnote

Field	Data Type	Reference	Remarks
active_flag	INTEGER NOT NULL	Boolean_Tabl e::enum	0=inactive, 1=active
close_date	LOCAL_TIME		When closed
cnote_type	INTEGER		announcement type
control_group	UUID		group to display to
del	INTEGER		Deleted flag (0=active 1=inactive/mark as deleted)
id	INTEGER UNIQUE NOT NULL KEY		Unique (to the table) Numeric ID
internal	INTEGER		Internal Flag
loc_id	UUID	ca_location::location_uuid	pointer to location

NOTIFICATION Table

Field	Data Type	Reference	Remarks
organization	UUID	ca_organization::uuid	pointer to Organization
persid	STRING 30		Persistent ID (SystemObjectName:id)
posted_by	UUID	ca_contact::uuid	who did the posting
posted_date	LOCAL_TIME		Last modify time
text	STRING 4000		message text

NOTIFICATION Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—NOTIFICATION
- **Object**—NOTIFICATION

Field	Data Type	Reference	Remarks
ALT_EMAIL	STRING 100		
ANALYST_ID	UUID	ca_contact::uuid	
DOC_ID	INTEGER		
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
NTF_LEVEL	INTEGER		0 - External and Internal 1 - External only 2 - Internal only

Notification_Urgency Table

Maps internal enums to strings for representing notification urgency for contacts.

- **SQL Name**—noturg
- **Object**—noturg

Field	Data Type	Index	Reference	Remarks
	INTEGER NOT_NULL		Active_ Boolean_ Table::enu m	0=present, 1=gone
desc	STRING 40			description of notification urgency
enum	INTEGER NOT_NULL			enumeration value
id	INTEGER UNIQUE NOT_NULL KEY	storage hash		key ID
sym	STRING 60 UNIQUE NOT_NULL S_KEY	sort dsc		notify urgency symbol

Notify_Log_Header Table

This table keeps track of each notify, and what happened to it.

- **SQL Name**—not_log
- **Object**—lr

Field	Data Type	Reference	Remarks
cmth_used	INTEGER	Contact_Met hod::id	
cntxt_obj	STRING 30		termination of sequence Context for notification
del	INTEGER NOT_NULL	Active_Boole an_Table::e num	Deleted flag (0=active 1=inactive/mark as deleted)

[Notify_Log_Header Table](#)

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod	LOCAL_TIME		Last modification time, for purging
nlh_ack_by	LOCAL_TIME		time deadline for ack.
nlh_ack_time	DURATION		How long for ack
nlh_c_address ee	UUID	ca_contact:: alias contact uuid	
nlh_c_alias	UUID		
nlh_cm_metho d	INTEGER	Notification_Urgency::enum	
nlh_email	STRING 50		Specifies the resolved email address (the email notification sent was not associated with the proper contact).
nlh_end	LOCAL_TIME		time of ACK or FYI final
nlh_hdr	STRING 40		Msg header text
nlh_msg	STRING 4000		message text
nlh_msg_html	STRING 32768		html version of notification if sent via mail
nlh_pri	INTEGER		generating notification enum priority of transition event
nlh_start	LOCAL_TIME NOT_NULL		Specifies the notification start date.
nlh_status	INTEGER		Pending send, sent FYI,
nlh_transition	INTEGER		notify method used - redefined in majic, points to 'noturg' object Transition point
nlh_type	INTEGER		FYI or ack
nlh_user_ack	STRING 40		sent ack, acked, nacked, cleared. Who acknowledged or cleared it.

Notify_Object_Attr Table

List of object attribute contacts that may receive notifications.

- **SQL Name**—ntfl
- **Object**—ntfl

Field	Data Type	Reference	Remarks
description	STRING 240		Textual description
del	INTEGER NOT_NULL	Active_Boolean_ Table::enum	Deleted flag (0=active 1=inactive/mark as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_b_y	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_d_t	LOCAL_TIME		Timestamp of when this record was last modified
object_attr	STRING 250		attr name in object
object_type	STRING 30		object name
persid	STRING 30		Persistent ID (SystemObjectName:id)
sym	STRING 30 NOT_NULL		

NR_Comment Table

Standard comments table.

- **SQL Name**—nr_com
- **Object**—nr_com

Field	Data Type	Referenc e	Remarks
attr_name	nvarchar(60)		For ITIL, this contains the Asset attribute that has changed.

O_COMMENTS Table

Field	Data Type	Reference	Remarks
com_comment	nvarchar(1000)		Identifies the comment text.
com_dt	INTEGER		Acts as the pointer to the parent row (nr_did), for the dt_comment written.
com_par_id	byte(16)	ca_owned_resource: :uuid	Foreign key to the id field of the ca_owned_resource table, this is the Asset.
com_userid	nvarchar(40)		Identifies the userid of the commenting author.
id	INTEGER		Primary key of this table, this is a unique numeric ID.
new_value	nvarchar(1000)		For ITIL, this contains the new value of the Asset's attribute that has changed.
old_value	nvarchar(1000)		For ITIL, this contains the old value of the Asset's attribute that has changed.
writer_id	byte(16)		Primary key to the table, this is a unique identifier.

O_COMMENTS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—O_COMMENTS
- **Object**—O_COMMENTS

Field	Data Type	Reference	Remarks
COMMENT_TEXT	STRING 255		
COMMENT_TIMESTAMP	LOCAL_TIME		
DOC_ID	INTEGER	SKELETONS::id	
EMAIL_ADDRESS	STRING 75		
ID	INTEGER KEY		Unique (to the table) Numeric ID

Field	Data Type	Reference	Remarks
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
USER_ID	UUID	ca_contact::uuid	
USER_NAME	STRING 50		
VER_COUNT	INTEGER		

O_EVENTS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—O_EVENTS
- **Object**—O_EVENTS

Field	Data Type	Reference	Remarks
ACTION	STRING 32768		
ENTITY_ID	INTEGER	SKELETONS::id	
EVENT_NAME	STRING 50		
EVENT_TIMESTAMP	LOCAL_TIME P		
ID	INTEGER KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
VER_COUNT	INTEGER		
WF_ACTION_ID	INTEGER		
WF_USER_ID	UUID	ca_contact::uuid	

O_INDEXES Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—O_INDEXES
- **Object**—KCAT

Field	Data Type	Reference	Remarks
DESCRIPTION	STRING 255		Textual description
AUTHOR_ID	UUID	ca_contact::uuid	
CAPTION	STRING 50		
DOC_TEMPLATE	INTEGER	CI_DOC_TEMPLATES::id	
HAS_CHILDREN	INTEGER		
HAS_DOCS	INTEGER		
ID	INTEGER KEY		Unique (to the table) Numeric ID
KEYWORDS	STRING 255		
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
OWNER_ID	UUID	ca_contact::uuid	
PARENT_ID	INTEGER	O_INDEXES::id	
PERMISSION_IN	INTEGER	O_INDEXES::id	
DEX_ID			
READ_PGROUP	INTEGER	P_GROUPS::id	
RELATIONAL_ID	STRING 255		
SUBJECT_EXPER	UUID	ca_contact::uuid	
T_ID			
WF_TEMPLATE	INTEGER	CI_WF_TEMPLATE_S::id	
WRITE_PGROUP	INTEGER	P_GROUPS::id	

Options Table

One row for every options added to call manager or any other BOP related stuff.

- **SQL Name**—options
- **Object**—options

Field	Data Type	Reference	Remarks
description	STRING 200		Textual description
action	INTEGER		
action_status	STRING 20		In Progress Installed Not Installed Install Failed Deinstall Failed
app_name	STRING 30		
default_value	STRING 100		
deinstall_script	STRING 200		
del	INTEGER NOT_NULL	Active_Boolean_T able::enum	Deleted flag (0=active 1=inactive/mark as deleted)
error_msg	STRING 100		
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
install_script	STRING 200		
last_mod_by	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
option_name	STRING 30		Option Name
persid	STRING 30		Persistent ID (SystemObjectName:id)
readme	STRING 200		
sequence	INTEGER		Sequence number for grouping and GUI

P_GROUPS Table

Field	Data Type	Reference	Remarks
sym	STRING 30 NOT_NULL		
validation	STRING 100		
value	STRING 100		
value_active	INTEGER		1=Yes 0=No

P_GROUPS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—P_GROUPS
- **Object**—P_GROUPS

Field	Data Type	Reference	Remarks
GRP_LIST	STRING 4096		
GRP_LIST_KEY	STRING 255		
ID	INTEGER KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified

Pcat_Group Table

Used to build a list of Groups that can service the Category.

- **SQL Name**—pcat_grp
- **Object**—pcat_grp

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		left hand attribute name
l_persid	STRING 60		left hand key
l_sql	INTEGER		left hand sort order

Field	Data Type	Reference	Remarks
r_attr	STRING 30		right hand attribute name
r_persid	STRING 60		right hand key
r_sql	INTEGER		right hand sort order

Pcat_Loc Table

The following tables are added to support many-to-many relationships between Categories, Groups that can service the Categories, and Locations that groups are able to service. This is all used for the Auto-Assignment functionality. Used to build a list of Service Locations valid for the Category.

- **SQL Name**—pcat_loc
- **Object**—pcat_loc

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		left hand attribute name
l_persid	STRING 60		left hand key
l_sql	INTEGER		left hand sort order
r_attr	STRING 30		right hand attribute name
r_persid	STRING 60		right hand key
r_sql	INTEGER		right hand sort order

Pcat_Workshift Table

Used to build the list of valid service workshifts for the Category.

- **SQL Name**—pcat_wrkshft
- **Object**—pcat_workshift

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		left hand attribute name

Field	Data Type	Reference	Remarks
l_persid	STRING 60		left hand key
l_sql	INTEGER		left hand sort order
r_attr	STRING 30		right hand attribute name
r_persid	STRING 60		right hand key
r_sql	INTEGER		right hand sort order

Person_Contacting Table

Reference table to denote the type of customer that made the contact. For example, the consumer, lawyer, media, and so on.

- **SQL Name**—perscon
- **Object**—perscnt

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL	Active_ Boolean_ Table::enum	Deleted flag that indicates the following: 0=Active 1=Inactive/marketed as deleted
id	INTEGER		Primary key of this table.
last_mod _by	byte(16) ::uuid	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod _dt	INTEGER		Specifies the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id)
sym	nvarchar(60)		The symbolic value for this Person_Contacting.

Priority Table

List of Priority entries. The priority reflects the time-frame in which a ticket must be resolved. For the ticket, it represents the highest priority of any problem attached to the ticket. Problem priorities are derived from the scope (impact) and severity of the problem.

- **SQL Name**—pri
- **Object**—pri

Field	Data Type	Reference	Remarks
del	INTEGER	Active_ Boolean_ Table::enum	Deleted flag that indicates the following: 0=Active 1=Inactive/marketed as deleted
enum	INTEGER		Primary key of this table.
id	INTEGER		Specifies the unique (to the table) numeric ID.
last_mod_ by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_ dt	INTEGER		Specifies the timestamp of when this record was last modified.
nx_desc	nvarchar(40)		Describes the priority.
service _type	nvarchar(30)	Service_ Desc ::code	Classic Service Type. Foreign key to the code field of the srv_desc table.
sym	nvarchar(12)		Indicates the symbolic name for this priority.

Prob_Category Table

Call Request call areas. Category of the issue the customer is calling about. May be hierarchical.

- **SQL Name**—prob_ctg
- **Object**—pcat

Field	Data Type	Reference	Remarks
assignee	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is the Assignee.
auto_assign	INTEGER		Represents the flag that enables auto assignment.
cr_flag	INTEGER		Represents the cr_flag status. When set to 1, this status is valid for requests.
del	INTEGER	Active_Boolean _Table::enum	Delete flag that indicates the following: 0=Active 1=Inactive/mark as deleted
description	nvarchar(500)		Identifies the textual description of the call area.
group_id	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is the Group.
id	INTEGER		Unique (to the table) numeric ID.
in_flag	INTEGER		Specifies the Incident flag. When set to 1, the status is valid for Incidents.
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.

Field	Data Type	Reference	Remarks
organization	byte(16)	ca_organization::organization_uuid	Foreign key to the id field of the ca_organization table, this is the Organization.
owning_contract	INTEGER	Service_Contract::id	Foreign key to the id field of the svc_contract table. This is the Service Contract.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id)
pr_flag	INTEGER		Specifies the Problem flag. When set to 1, the status is valid for Problems.
schedule	INTEGER		Deprecated.
service_type	nvarchar(30)	Service_Desc::code	Classic Service Type. Foreign key to the code field of the srv_desc table, this is the Classic Service Type.
survey	INTEGER	Survey_Template::id	Foreign key of the id field of the survey_tpl table, this is the Survey.
sym	nvarchar(30)		Specifies the Request Area's symbolic name.
tcode	INTEGER		Deprecated.

Product Table

Reference table to denote the product that the complaint relates to.

- **SQL Name**—product
- **Object**—prod

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	Delete flag that indicates the following: 0=Active 1=Inactive/marketed as deleted

Field	Data Type	Reference	Remarks
id	INTEGER		Primary key of this table.
last_mod_by	byte(16)	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sym	nvarchar(60)		Specifies the symbolic value for this Product.

Property Table

Property value pairs for an object.

- **SQL Name**—prp
- **Object**—prp

Field	Data Type	Reference	Remarks
description	nvarchar(240)		Identifies the textual description of this property.
id	INTEGER		Primary key of this table.
label	nvarchar(80)		Identifies the label value for this property.
last_mod_by	byte(16)	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
object_id	INTEGER	chg id	Identifies the unique (to the table) numeric ID.
object_type	nvarchar(30)		Specifies the short name of the object to which this property belongs.

Field	Data Type	Reference	Remarks
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
property	INTEGER	Property_Template ::id	Foreign key to the id field of the prptpl table, this is the Template.
required	INTEGER	Boolean _Table ::enum	Specifies the Required field as follows: 0=Not required 1=Required
sample	nvarchar(240)		Indicates the sample value for this property.
sequence	INTEGER		Specifies the sequence value for this property.
value	nvarchar(240)		Specifies the value of the property.

Property_Template Table

Additional properties for objects.

- **SQL Name**—prptpl
- **Object**—prptpl

Field	Data Type	Reference	Remarks
code	nvarchar(12)		Specifies the non-editable handle for the query.
del	INTEGER	Active_ Boolean _Table::: enum	Deleted flag that indicates the following: 0=Active 1=Inactive/marketed as deleted
description	nvarchar(240)		Specifies the textual description for the property template.
label	nvarchar(80)		Display the text for the property.
last_mod_by	byte(16)	ca_ contact ::uuid	Specifies the UUID of the contact who last modified this record.

Field	Data Type	Reference	Remarks
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
object_attrname	nvarchar(30)		Specifies that this template belongs to the attribute name in the object.
object_attrval	INTEGER		Identifies the object attribute value, which drives the template attribute value.
object_type	nvarchar(30)		Specifies the short name of the object.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
prptbl_id			Indicates the unique (to the table) numeric ID.
required	INTEGER NOT_NULL		Identifies the Required flag as follows: 0=Not Required 1=Required
sample	nvarchar(240)		Provides a sample, or Help text.
sequence	INTEGER		Specifies the display order.

Queued_Notify Table

Notifications that are queued due to workshifts are saved here.

- **SQL Name**—not_que
- **Object**—notque

Field	Data Type	Reference	Remarks
cmth_override	INTEGER		method over ride
context_persid	STRING 30		persistent id of object for notification

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL	Active_ Boolean_ Table::enum	Deleted flag (0=active 1=inactive/marketed as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
internal	INTEGER		internal notification
msg_ack	STRING 40		message acknowledgment
msg_body	STRING 1000		message text
msg_body_html	STRING 32768		message text
msg_title	STRING 40		Msg header text
notify_level	INTEGER		notification level
persid	STRING 30		Persistent ID (SystemObjectName:id)
transition_pt	INTEGER		transition point

Quick_Template_Types Table

Quick_Template_Types - Reference table for quick template types.

- **SQL Name**—quick_tpl_types
- **Object**—quick_tpl_types

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL	Active_Boolean _Table::enum	Deleted flag (0=active 1=inactive/marketed as deleted)
enum	INTEGER UNIQUE NOT_NULL		Enumerated value for this entry - specifies ordering in lists and relative values
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
nx_desc	STRING 40		Descriptive Info

Field	Data Type	Reference	Remarks
persid	STRING 30		Persistent ID (SystemObjectName:id)
sym	STRING 30		Symbolic name of level

Remote_Ref Table

Remote References. Used for smart hooks. Determines what command to execute. Different command for unix and pc's using the same smart hook. Can apply security to smart hook.

- **SQL Name**—rem_ref
- **Object**—rrf

Field	Data Type	Reference	Remarks
description	STRING 500		The description of the command.
arch_type	STRING 12		architecture to exec this on unix, pc. if empty, then all.
code	STRING 12 UNIQUE NOT_NULL S_KEY		noneditable key for command
del	INTEGER NOT_NULL	Active_ Boolean_ Table::enum	Deleted flag (0=active 1=inactive/mark as deleted)
exec_str	STRING 500		string to execute on unix
function_group	STRING 30		function group for security
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_by	UUID	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
pcexec_str	STRING 500		string to execute on pc
sym	STRING 30 NOT_NULL		name of command

Reporting_Method Table

Reference table to denote how the contact with the customer occurred.
Example: email, phone.

- **SQL Name**—repmeth
- **Object**—rptmeth

Field	Data Type	Reference	Remarks
del	INTEGER	Active_ Boolean_ Table::: enum	Specifies the Deleted flag as follows: 0=Active 1=Inactive/marketed as deleted
id	INTEGER		Primary key of this table.
last_mod_by	byte(16)	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Indicates the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sym	nvarchar(60)		Identifies the symbolic value for this Reporting Method.

Req_Property Table

Property value pairs for a Request.

- **SQL Name**—cr_prp
- **Object**—cr_prp

Field	Data Type	Reference	Remarks
description	nvarchar(240)		Specifies the textual description of this property.
id	INTEGER		Primary key of this table, it is a unique, numeric ID.

Req_Property_Template Table

Field	Data Type	Reference	Remarks
label	nvarchar(80)		Specifies the label value for this Request property.
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
owning_cr	nvarchar(30)	Call_Req ::persid	Persistent ID (SystemObjectName:id)
required	INTEGER		Identifies the Required flag, as follows: 0=Not required 1=Required
sample	nvarchar(240)		The sample value for this Request_Property.
sequence	INTEGER		The sequence value for this Request property.
value	nvarchar(240)		Identifies the value of the property.

Req_Property_Template Table

Templates used to specify request properties.

- **SQL Name**—cr_prptpl
- **Object**—cr_prptpl

Field	Data Type	Reference	Remarks
description	STRING 240		Textual description of this template
code	STRING 12 UNIQUE NOT_NULL S_KEY		noneditable handle for query
del	INTEGER NOT_NULL	Active_ Boolean _Table::: enum	Deleted flag (0=active 1=inactive/mark as deleted)

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
label	STRING 80 NOT_NULL		display text for property
last_mod_by	UUID	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
owning_area	STRING 30 NOT_NULL	Prob_ Category ::persid	parent category
persid	STRING 30		Persistent ID (SystemObjectName:id)
required	INTEGER NOT_NULL		0 = not required
sample	STRING 240		help text
sequence	INTEGER NOT_NULL		display order

Response Table

Personalized response text used to simplify data entry when using the Unicentser Service Desk applications.

- **SQL Name**—response
- **Object**—response

Field	Data Type	Reference	Remarks
chg_flag	INTEGER S_KEY		
cr_flag	INTEGER S_KEY		
del	INTEGER NOT_NULL	Active_Boolean_ Table::enum	Deleted flag (0=active 1=inactive/marketed as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID

Reverse_Boolean_Table Table

Field	Data Type	Reference	Remarks
in_flag	INTEGER S_KEY		
iss_flag	INTEGER S_KEY		
last_mod_by	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
persid	STRING 30		Persistent ID (SystemObjectName:id)
pr_flag	INTEGER S_KEY		
response	STRING 1000		response text
response_owner	UUID S_KEY	ca_contact::uuid	response owner
sym	STRING 50 NOT_NULL S_KEY		symbol

Reverse_Boolean_Table Table

Reverse boolean lookup table.

- **SQL Name**—rboottab
- **Object**—rev_bool

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL		Deleted flag (0=active 1=inactive/mark as deleted)
enum	INTEGER NOT_NULL		Enumerated value for this entry - specifies ordering in lists and relative values
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
nx_des	STRING 40		c

Field	Data Type	Reference	Remarks
sym	STRING 12 UNIQUE NOT_NULL S_KEY		

Rootcause Table

Reference table to denote the rootcause type used when resolving or closing a request, change order, or issue.

- **SQL Name**—rootcause
- **Object**—rc

Field	Data Type	Reference	Remarks
del	INTEGER	Active_ Boolean_ Table::: enum	Specifies the Deleted flag as follows: 0=Active 1=Inactive/marketed as deleted
description	nvarchar(240)		Provides a textual description of this root cause.
id	INTEGER		Primary key of this table.
last_mod_by	byte(16)	ca_contact :: uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Indicates the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sym	nvarchar(60)		Identifies the symbolic value for this Rootcause.

Rpt_Meth Table

Reporting methods used to display information within the Unicenter Service Desk applications.

- **SQL Name**—rptmth
- **Object**—rptm

Field	Data Type	Reference	Remarks
description	STRING 80		Textual description of this reporting method
def_pg_len	STRING 80		page length default
default_out	STRING 80		output default
del	INTEGER NOT_NULL	Active_ Boolean_ Table::: enum	Deleted flag (0=active 1=inactive/mark as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
is_default	INTEGER		If set, this is the default reporting method.
last_mod_by	UUID	ca_contact:: uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
script	STRING 1000		
sym	STRING 30 NOT_NULL		

SA_Policy Table

Policy information for dealing with automated creation and access of Unicenter Service Desk components.

- **SQL Name**—sapolicy
- **Object**—sapolicy

Field	Data Type	Reference	Remarks
description	STRING 300		policy description
access_atmnt	INTEGER		number of attachment per hour
access_data	INTEGER		number of query operations per hour
access_knowl edge	INTEGER		number of knowledge related operations per hour
access_object _ins	INTEGER		number of object insertions per hour
access_object _upd	INTEGER		number of object updates per hour
access_ticket _ins	INTEGER		number of ticket insertions per hour
allow_impers onate	INTEGER NOT_NULL	Active_Boolean _Table::enum	0=not allowed, 1=allowed
code	STRING 20 UNIQUE NOT_NULL S_KEY		unique code name
del	INTEGER NOT_NULL	Active_Boolean _Table::enum	Deleted flag (0=active 1=inactive/mark as deleted)
ext_appl	INTEGER		Application using policy
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
is_default	INTEGER		default flag
last_mod_by	UUID	ca_contact::uui d	Specifies the UUID of the contact who last modified this record

[SA_Prob_Type Table](#)

Field	Data Type	Reference	Remarks
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
persid	STRING 30		Persistent ID (SystemObjectName:id)
proxy_contact	UUID	ca_contact::uuid	default contact
pub_key	STRING 2000		public key in BASE64
state	INTEGER		checkin state(for tracking purpose)
sym	STRING 40 NOT_NULL		symbolic name of policy

[SA_Prob_Type Table](#)

Problem type definitions used with automated creation policies within Unicenter Service Desk applications.

- **SQL Name**—saprobtyp
- **Object**—saprobtyp

Field	Data Type	Reference	Remarks
description	STRING 300		Textual description
code	STRING 20 UNIQUE NOT_NULL S_KEY		unique code name
del	INTEGER NOT_NULL	Active_Boolean an_Table::en	Deleted flag (0=active 1=inactive/mark as deleted)
dup_action	INTEGER		enum for action to handle ticket duplication
dup_interval	DURATION		time range for searching duplicate tickets
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID

Field	Data Type	Reference	Remarks
is_default	INTEGER		one and only one for all policies.
is_internal	INTEGER	Active_Boole an_Table::en um	OTB problem types cannot be deleted.
last_mod_b y	UUID	ca_contact:: uuid	Specifies the UUID of the contact who last modified this record
last_mod_d t	LOCAL_TI ME		Timestamp of when this record was last modified
owning_poli cy	INTEGER	SA_Policy::id	the owner of this problem type
persid	STRING 30		Persistent ID (SystemObjectName:id)
ret_app_1	STRING 500		text passed back to program for ticket creation.
ret_usr_1	STRING 500		text passed back to human for ticket creation.
sym	STRING 40 NOT_NULL		symbolic name of problem type
ticket_tmpl _fac	STRING 20		factory of the ticket template
ticket_tmpl _id	INTEGER		id of the ticket template
ticket_tmpl _name	STRING 40		name of the ticket template

Sequence_Control Table

Used to determine what to use for a prefix and suffix when generating call request numbers. Users may not create new records or delete records in this table.

- **SQL Name**—seqctl
- **Object**—seq

Field	Data Type	Reference	Remarks
description	STRING 240		Textual description

Field	Data Type	Reference	Remarks
code	STRING 12 UNIQUE NOT_NULL S_KEY		nonditable key for record
del	INTEGER NOT_NULL	Active_Bool ean_Table::enum	Deleted flag (0=active 1=inactive/mark as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_by	UUID	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
prefix	STRING 5		prefix to sequence number
suffix	STRING 5		suffix to sequence number
sym	STRING 30 NOT_NULL		

Server_Aliases Table

Contain server alias names that are valid for various server-client zones.

- **SQL Name**—srvr_aliases
- **Object**—srvr_aliases

Field	Data Type	Reference	Remarks
alias_name	STRING 30 NOT_NULL		Alias name
del	INTEGER NOT_NULL	Active_Boolean_T able::enum	Deleted flag (0=active 1=inactive/mark as deleted)
host_addr	STRING 30		Translated Host Address
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID

Field	Data Type	Reference	Remarks
last_mod_by	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
zone_id	INTEGER NOT_NULL	Server_Zones::id	Zone name

Server_Zones Table

Contain regional server zone names.

- **SQL Name**—srvr_zones
- **Object**—srvr_zones

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL	Active_Boolean_Table::enum	Deleted flag (0=active 1=inactive/mark as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
is_default	INTEGER	Boolean_Table::e 1 default zone num	
last_mod_by	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
zone_name	STRING 30 NOT_NULL		Zone name

Service_Contract Table

Used to track relationships between orgs, ticket attr's and svc types. Used for SLA management.

- **SQL Name**—svc_contract
- **Object**—svc_contract

Field	Data Type	Reference	Remarks
active	INTEGER	Active_Boolean _Table::enum	Specifies the status of the contract, as follows: 0=Inactive 1= Active
contract_num	nvarchar(50)		Identifies the Contract ID.
del	INTEGER	Active_Boolean _Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marked as deleted
dflt_chgcat_st	nvarchar(30)	Service_Desc ::code	Defines the default service type for the change category.
dflt_cnt_st	nvarchar(30)	Service_Desc ::code	Defines the default service type for the end user.
dflt_isscat_st	nvarchar(30)	Service_Desc ::code	Defines the default service type for the issue category.
dflt_nr_st	nvarchar(30)	Service_Desc ::code	Defines the default service type for the asset.
dflt_pcat_st	nvarchar(30)	Service_Desc ::code	Defines the default service type for the request area.
dflt_pri_st	nvarchar(30)	Service_Desc ::code	Defines the default service type for the priority.
expiration	INTEGER		Specifies the Contract expiration date.
id	INTEGER		Primary key of this table.
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.

Field	Data Type	Reference	Remarks
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
nx_desc	nvarchar(240)		Specifies descriptive information.
org_svc_type	nvarchar(30)	Service_Desc ::code	Defines the service type for the organization.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
svc_advocate	byte(16)	ca_contact ::uuid	Identifies the customer advocate for the organizations assigned to the contract.
svc_owner	byte(16)	ca_contact ::uuid	Foreign key to the contact_uuid field of the ca_contact table, this is the service desk person responsible for contract.
sym	nvarchar(80)		Specifies the symbolic name of the contract level.

Service_Desc Table

This table contains the Call Request Service Types that are not related to service level agreements in Problem Manager. These may be used to associate the types of service call requests to receive, and are can be defined by the user. Examples include: Platinum, Gold, Silver, or Bronze.

- **SQL Name**—srv_desc
- **Object**—sdsc

Field	Data Type	Reference	Remarks
code	nvarchar(30)		Primary key of this table.
del	INTEGER	Active_Boolean _Table::enum	Specifies the status of the Deleted flag as follows: 0=Active 1=Inactive/marketed as deleted

Field	Data Type	Reference	Remarks
description	nvarchar (500)		Provides the text description of service.
id	INTEGER		Primary key of this table.
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp for when this record was last modified.
owning_contract	INTEGER	Service_Contract::id	Specifies the unique (to the table) numeric ID.
persid	nvarchar(30)		Identifies the Persistent ID: (SystemObjectName:id)
rank	INTEGER		Identifies the ranking status of the service type to determine level of priority.
schedule	nvarchar(30)	Bop_Workshift ::persid	Foreign key to the persistent_id field of the bpwsht table, this is the Schedule.
sym	nvarchar(30)		Identifies the name of the service type.
violation_cost	INTEGER		Specifies the monetary cost for violation.

session_log Table

Session log.

- **SQL Name**—session_log
- **Object**—session_log

Field	Data Type	Reference	Remarks
contact	byte(16)	ca_contact ::uuid	Foreign key to the contact_uuid field of the ca_contact table, this is the User.

Field	Data Type	Reference	Remarks
id	INTEGER		Primary key of this table
login_time	INTEGER		Indicates the time of when the session began.
logout_time	INTEGER		Indicates the time of when the session ended.
policy	INTEGER	SA_Policy :::id	Specifies the session policy.
session_id	INTEGER		Displays the ID if the status is okay.
session_type	INTEGER	session_type :::id	Specifies the unique (to the table) numeric ID.
status	INTEGER		Identifies the Login status: 0=Okay

session_type Table

Session type (Web client, Java client, and so on).

- **SQL Name**—session_type
- **Object**—session_type

Field	Data Type	Reference	Remarks
description	nvarchar(500)		Identifies the description of the session type.
id	INTEGER		Primary key of this table.
last_mod_by	byte(16) uuid	ca_contact:: uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id)
sym	nvarchar(30)		Identifies the name of the Session type.

Severity Table

List of severity definitions used by Unicenter Service Desk applications.

- **SQL Name**—sevrty
- **Object**—sev

Field	Data Type	Reference	Remarks
del	INTEGER	Active_ Boolean_ Table:::enum	Identifies the Deleted flag as follows: 0=Active 1=Inactive/marketed as deleted
enum	INTEGER		Primary key of this table.
id	INTEGER		Unique (to the table) numeric ID.
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
nx_desc	nvarchar(40)		Describes the severity.
sym	nvarchar(12)		Identifies the symbolic name for this severity.

SHOW_OBJ Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—SHOW_OBJ
- **Object**—SHOW_OBJ

Field	Data Type	Reference	Remarks
EXPIRE_DATE	LOCAL_TIME		
ID	INTEGER KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified

Field	Data Type	Reference	Remarks
OBJ_PERSID	STRING 255		Persistent ID (SystemObjectName:id)
PWD	STRING 255		

SKELETONS Table

Program control table used by Unicenter Knowledge Tools.

- **SQL Name**—KD
- **Object**—SKELETONS

Field	Data Type	Reference	Remarks
ACTIVE_STATE	INTEGER		
ACTIVE_STATE	LOCAL_TIME		
_DATE			
ASSIGNEE_ID	UUID	ca_contact:: uuid	
AUTHOR_ID	UUID	ca_contact:: uuid	
BU_RESULT	REAL		
CREATED_VIA	INTEGER		
CREATION_DA	LOCAL_TIME		The timestamp indicating when this record was created
TE			
CURRENT_ACT	INTEGER	CI_ACTIONS	
ION_ID		::id	
CUSTOM1	STRING 50		
CUSTOM2	STRING 50		
CUSTOM3	STRING 50		
CUSTOM4	STRING 255		
CUSTOM5	STRING 255		
CUSTOM_NUM	REAL		
1			
CUSTOM_NUM	REAL		
2			

SKELETONS Table

Field	Data Type	Reference	Remarks
DOC_TEMPLAT E_ID	INTEGER	CI_DOC_TE MPLATES::id	
DOC_TYPE_ID	INTEGER	CI_DOC_TYP ES::id	
DOC_VERSION	STRING 50		
EXPIRATION_D ATE	LOCAL_TIME		
EXPIRE_NOTIF ICATION_SENT	INTEGER		
EXT_DOC_ID	INTEGER		
FULLWORDS	STRING 32768		
HITS	INTEGER		
ID	INTEGER KEY		Unique (to the table) Numeric ID
INDEXED	INTEGER		
INHERIT_PERM ISSION	INTEGER		
INITIATOR	STRING 100		
INITIATOR_ID	UUID	ca_contact:: uuid	
KD_PERMISSI ON_INDEX_ID	INTEGER	O_INDEXES: :id	
LAST_ACCEPTE D_DATE	LOCAL_TIME		
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
LOCKED_BY_I D	UUID	ca_contact:: uuid	
MODIFY_DATE	LOCAL_TIME		
NOTES	STRING 32768		
OWNER_ID	UUID	ca_contact:: uuid	
PARENT_CR	STRING 30	Call_Req::pe rsid	

Field	Data Type	Reference	Remarks
PARENT_ISS	STRING 30	issue persistent_id	
PRIMARY_IND	INTEGER	O_INDEXES: EX	:id
PRIORITY_ID	INTEGER	CI_PRIORITY ES::id	
PROBLEM	STRING 32768		
PUBLISHED_DATE	LOCAL_TIME		
READ_PGROUP	INTEGER	P_GROUPS::i d	
RESOLUTION	STRING 32768		
RESOLUTION_LENGTH	INTEGER		
RESOLUTION_SHORT	STRING 255 SHORT		
REVIEW_DATE	LOCAL_TIME		
SD_ACCEPTED_HITS	INTEGER		
SD_ASSET_ID	UUID	ca_owned_re source::uuid	
SD_IMPACT_ID	INTEGER	Impact::enu m	
SD_PRIORITY_ID	INTEGER	Priority::enu m	
SD_PRODUCT_ID	INTEGER	Product::id	
SD_ROOTCAUSE_ID	INTEGER	Rootcause::i d	
SD_SEVERITY_ID	INTEGER	Severity::en um	
SD_URGENCY_ID	INTEGER	Urgency::en um	
SHORTWORDS	STRING 32768		

SKELETONS Table

Field	Data Type	Reference	Remarks
START_DATE	LOCAL_TIME		
STATUS_ID	INTEGER	CI_STATUSE S::id	
SUBJECT_EXPE RT_ID	UUID	ca_contact:: uuid	
SUMMARY	STRING 255		
TITLE	STRING 255		
USER_DEF_ID	STRING 40		
User_slv_cnt	INTEGER		
Vote_count	INTEGER		
Avg_rating	FLOAT		
Faq_sym	INTEGER		
VER_COMMEN T	STRING 1000		
VER_COUNT	INTEGER		
VER_CROSS_R EF_ID	INTEGER		
WF_TEMPLATE	INTEGER	CI_WF_TEMP LATES::id	
WORD_COUNT _TOTAL	INTEGER		
WORDCOUNT	INTEGER		
WORDCOUNTS	STRING 32768		
WORDORDERS	STRING 32768		
WORDPLACES	STRING 32768		
WORDSPANS	STRING 32768		
WRITE_PGROU P	INTEGER	P_GROUPS::i d	
NOTES	STRING 32768		

SLA_Contract_Map Table

Maps a service type to a reference object; used by Service_Contract objects.

- **SQL Name**—sdsc_map
- **Object**—sdsc_map

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_by	UUID	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
map_contract	INTEGER NOT_NULL	Service_Co ntract::id	
map_persid	STRING 60		
map_sdsc	STRING 30	Service_De sc::code	
persid	STRING 30		Persistent ID (SystemObjectName:id)

SLA_Template Table

Service Level Agreement Templates for Call and Change. Not related to service level agreements in problem manager. Links Service Descriptions with Events.

- **SQL Name**—slatpl
- **Object**—slatpl

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL	Active_Bool ean_Table:: enum	Deleted flag (0=active 1=inactive/marketed as deleted)
elapsed	DURATION NOT_NULL		elapsed time to delay event

Field	Data Type	Reference	Remarks
event	STRING 30	Events::per sid	what to do
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_by	UUID	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
object_type	STRING 30		kind of object for this slai (cr, chg)
persid	STRING 30		Persistent ID (SystemObjectName:id)
service_type	STRING 30 NOT_NULL	Service_De sc:::code	
sym	STRING 30 NOT_NULL S_KEY		name of sla

Spell_Macro Table

This table stores proprietary Spell macros and macro fragments .

- **SQL Name**—splmac
- **Object**—macro

Field	Data Type	Reference	Remarks
description	STRING 80		Identifies the text description.
del	INTEGER NOT_NULL	Active_Boolean_Table:::enum	Specifies the Deleted flag status, for example, 0=active and 1=inactive/marketed as deleted).
fragment	STRING 4000		
id	INTEGER UNIQUE NOT_NULL KEY		Identifies the Numeric ID that is unique to the table.

Field	Data Type	Reference	Remarks
last_mod_dt	LOCAL_TIME		Specifies the Timestamp for when this record was last modified.
lock_object	INTEGER NOT_NULL		Specifies the boolean of the object.
msg_html	STRING 32768		Identifies the message used to keep the html template.
ob_type	STRING 30 NOT_NULL		
persid	STRING 30		Identifies the Persistent ID: (SystemObjectName:id)
sym	STRING 30 NOT_NULL UNIQUE NOT_NULL		
type	STRING 30 NOT_NULL	Spell_Macro_Type::persid	
usr_integer_1	INTEGER		
usr_integer_2	INTEGER		
usr_integer_3	INTEGER		
usr_string2	STRING 250		
usr_string3	STRING 125		
usr_string4	STRING 25		

Spell_Macro_Type Table

Stores type information for proprietary Spell macros and macro fragments .

- **SQL Name**—splmactp
- **Object**—macro_type

Field	Data Type	Reference	Remarks
description	STRING 200		Textual description
arg_list	STRING 80		
code	STRING 30 UNIQUE NOT_NULL NOT_NULL		
del	INTEGER NOT_NULL	Active_Boolean _Table::enum	Deleted flag (0=active 1=inactive/marketed as deleted)
display_name	STRING 30		form name to display
execute_scrip t	STRING 800		
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
lock_object_fl ag	INTEGER		
persid	STRING 30		Persistent ID (SystemObjectName:id)
sym	STRING 30 NOT_NULL		
tech_desc	STRING 300		
validate_scrip t	STRING 400		

SQL_Script Table

Stores SQL scripts used by the Unicenter Service Desk applications.

- **SQL Name**—sql_tab

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL		Deleted flag (0=active 1=inactive/mark as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
sql_desc	STRING 240		user title for script
sql_name	STRING 30 S_KEY		user name for script
sql_script	STRING 500		actual script

Survey

Customer Survey.

- **SQL Name**—survey
- **Object**—survey

Field	Data Type	Reference	Remarks
comment_label	nvarchar(80)		The comment label value for this Survey.
conclude_text	nvarchar(400)		Indicates the text to display after the survey is completed.
del	INTEGER	Active_ Boolean_ Table::: enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/mark as deleted
description	nvarchar(400)		Specifies the textual description for this survey.

Field	Data Type	Reference	Remarks
id	INTEGER		Primary key of this table.
include_comment	INTEGER		Specifies the Include Comment flag, as follows: 0=No comment allowed
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
nx_comment	nvarchar(200)		Identifies the comment value for this Survey.
object_id	INTEGER		Identifies the ID of the object for this survey.
object_type	nvarchar(10)		Identifies the object type for this survey, for example, CR, CHG, and so on.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sym	nvarchar(12)		Identifies the symbolic name of this survey.

[Survey_Answer](#)

Customer Survey Answer.

- **SQL Name**—survey_answer
- **Object**—svy_ans

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
id	INTEGER		Primary key of this table.

Field	Data Type	Reference	Remarks
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
own_srvy_qu estion	INTEGER	Survey_Question::id	Unique (to the table) numeric ID.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
selected	INTEGER		Identifies the selected value for this Survey_Answer.
sequence	INTEGER		Specifies the display order of the survey answers.
txt	nvarchar(400)		Specifies the question text.

Survey_Answer_Template

Customer Survey Answer Template.

- **SQL Name**—survey_atpl
- **Object**—svy_atpl

Field	Data Type	Reference	Remarks
atbl_id	INTEGER		Identifies the unique (to the table) numeric ID.
del	INTEGER	Active_ Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.

Field	Data Type	Reference	Remarks
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
own_srvy_q	INTEGER	Survey_Question_Template::id	Unique (to the table) numeric ID.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sequence	INTEGER		Specifies the display order of the survey answers.
txt	nvarchar(400)		Identifies the answer text.

[Survey_Question Table](#)

Customer Survey Question.

- **SQL Name**—survey_question
- **Object**—svy_ques

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
id	INTEGER		Primary key of this table.
include_qcomment	INTEGER		Specifies the Include Question Comment flag, as follows: 1=Include comment box for this question
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.

Field	Data Type	Reference	Remarks
mult_resp_flag	INTEGER		Specifies the Multiple Response flag, as follows: 0=Choose 1 response (radio) 1=Choose "N" (check boxes)
owning_survey	INTEGER	Survey_id	Specifies the unique (to the table) numeric ID.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
qcomment_label	nvarchar(80)		Specifies the label for the comment.
resp_required	INTEGER		Specifies the Required Response flag, as follows: 1=Respondents must answer the question
response	INTEGER		Specifies the sequence number of the response.
sequence	INTEGER		Specifies the display order.
txt	nvarchar(400)		Identifies the question text.

Survey_Question_Template Table

Customer Survey Question Template.

- **SQL Name**—survey_qtpl
- **Object**—svy_qtpl

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
id	INTEGER		Primary key of this table.
include_qcomment	INTEGER		Specifies the Include Question Comment flag, as follows: 1=Include comment box for this question

Survey_Stats Table

Field	Data Type	Reference	Remarks
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
mult_resp_flag	INTEGER		Specifies the Multiple Response flag, as follows: 0=Choose 1 response (radio) 1=Choose "N" (check boxes)
owning_survey	INTEGER	Survey_Template ::id	Specifies the unique (to the table) numeric ID.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
qcomment_label	nvarchar(80)		Specifies the label for the comment.
resp_requir_ed	INTEGER		Specifies the Required Response flag, as follows: 1=Respondents must answer the question
sequence	INTEGER		Specifies the display order of the survey questions.
txt	nvarchar(400)		Identifies the question text.

Survey_Stats Table

This table contains customer survey statistics.

- **SQL Name**—survey_statistics
- **Object**—svystat

Field	Data Type	Reference	Remarks
cyc_counter	INTEGER		Identifies the cycle counter to be compared against the cycle (described in the next field).

Field	Data Type	Reference	Remarks
cycle	INTEGER		Identifies the submission cycle.
del	INTEGER	Active_Boolean _Table: :enum	Specifies the status of the Deleted flag, as follows: 0=Active 1=Inactive/marked deleted)
eval_counter	INTEGER		Identifies the number of times called to evaluate for notification submission.
id	INTEGER		Primary key of this table.
persid	nvarchar(30)		Specifies the Persistent ID: (SystemObjectName: id).
sub_counter	INTEGER		Identifies the number of times notification submission was approved.
tplid	INTEGER	Survey_ Template: :id	Identifies the unique (to the table) numeric ID.

Survey_Template Table

Customer Survey Template.

- **SQL Name**—survey_tpl
- **Object**—svy_tpl

Field	Data Type	Reference	Remarks
comment_label	nvarchar(80)		Identifies the comment label value for this Survey_Template.
conclude_text	nvarchar(400)		Specifies the text to display after the survey has been completed.

[Survey_Template Table](#)

Field	Data Type	Reference	Remarks
cycle_counter	INTEGER		Indicates to keep a running count for Submit Cycle.
del	INTEGER	Active_Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
description	nvarchar(400)		Indicates the textual description for this template.
id	INTEGER		Primary key of this table.
include_comment	INTEGER		Specifies the Include Question Comment flag, as follows: 0=No comment allowed
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Indicates the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
submit_cycle	INTEGER		Indicates to send the survey every 'nth' time.
sym	nvarchar(12)		Identifies the symbolic name of the survey template.
tracking_flag	INTEGER		Specifies the flag to track responses.

Survey_Tracking Table

Table used to track status of managed surveys.

- **SQL Name**—survey_tracking
- **Object**—svytrk

Field	Data Type	Reference	Remarks
cntid	byte(16)	ca_contact::uuid	Primary key of this table.
del	INTEGER	Active_ Boolean_ Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
id	INTEGER		Primary key of this table, this is a unique numeric ID.
notif_dt	INTEGER		Specifies the time of the last notification.
object_id	INTEGER		Identifies the ID of the object for this survey.
object_type	nvarchar(10)		Indicates the object type for this survey, such as, CR, CHG, and so on.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
recv_dt	INTEGER		Identifies the time the completed survey was received.
status	INTEGER		Identifies the Status flag, as follows: 1=Survey submitted
tplid	INTEGER	Survey_ Template::id	Unique (to the table) numeric ID.

Table_Name Table

Table name list used by Unicenter Service Desk applications.

- **SQL Name**—tn

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL		Deleted flag (0=active 1=inactive/marketed as deleted)
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
tn_desc	STRING 240		description or table
tn_dflt	STRING 30		default external name
tn_name	STRING 30		user name for table
tn_sys	STRING 30 S_KEY		system name

Task_Status Table

Workflow task states. Possible states include: Wait, pending, approve, reject, and so on.

- **SQL Name**—tskstat
- **Object**—tskstat

Field	Data Type	Reference	Remarks
allow_accumulate	INTEGER		Identifies the Allow Accumulate flag, as follows: 0=Do not accumulate 1=Accumulate
allow_task_update	INTEGER		Specifies the Allow Task Update flag, as follows: 0=Cannot update 1=Can update
code	nvarchar(12)		Primary key of this table.

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
description	nvarchar(500)		Provides a text description of the status.
do_next_task	INTEGER		Sets the Do Next Task flag, as follows: 0>No 1=Yes
hold	INTEGER		Sets the Hold flag, as follows: 0=Start events 1=Stop events
id	INTEGER		Specifies the unique (to the table) numeric ID.
is_internal	INTEGER		Specifies the Internal flag, as follows: 0>No 1=Yes (do not display in most status selections).
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp of when this record was last modified.
no_update_msg	nvarchar(500)		Sets the No Update Message flag to No.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id)
sym	nvarchar(30)		Identifies the Task Status name.

Field	Data Type	Reference	Remarks
task_ complete	INTEGER		Sets the Task Complete, as follows: 0=No 1=Yes

Timespan Table

List of defined time-span elements used by applications to calculate hours-of-operation time calculations.

- **SQL Name**—tspan
- **Object**—tspan

Field	Data Type	Reference	Remarks
code	STRING 10 UNIQUE NOT_NULL		
end_day	STRING 5		
end_hour	STRING 5		
end_minute	STRING 5		
end_month	STRING 5		
end_year	STRING 5		
id	INTEGER UNIQUE KEY		Unique (to the table) Numeric ID
last_mod_by	UUID	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
nx_desc	STRING 240		
start_day	STRING 5		
start_hour	STRING 5		
start_minute	STRING 5		
start_month	STRING 5		
start_year	STRING 5		

Field	Data Type	Reference	Remarks
sym	STRING 30 UNIQUE NOT_NULL		
trigger_day	STRING 5		
trigger_hour	STRING 5		
trigger_minut e	STRING 5		
trigger_mont h	STRING 5		
trigger_year	STRING 5		

Timezone Table

This table contains the Timezones used in Unicenter Service Desk applications..

- **SQL Name**—tz
- **Object**—tz

Field	Data Type	Reference	Remarks
code	nvarchar(12)		Primary key of this table.
del	INTEGER	Active_ Boolean_ Table::: enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
description	nvarchar(500)		Identifies the text description of the timezone.
dst_delta	INTEGER		Specifies the delta seconds for daylight saving time.
end_abs _date	LOCAL_TIME		Identifies the <i>absolute</i> start date.
end_day	INTEGER		Represents the end day of the week, such as 0-6.
end_mon	INTEGER		Represents the ending month, such as 0-11 for the month of the year.

Field	Data Type	Reference	Remarks
end_pos	INTEGER		Represents the ending position, such as 0 for "First" or 1 for "Last".
gmt_delta	INTEGER		Represents the delta seconds from GMT.
id	INTEGER		Identifies the numeric ID, which is unique to the table.
last_mod_by	byte(16)	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp for when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
start_abs_date	INTEGER		Identifies the <i>absolute</i> start date.
start_day	INTEGER		Represents the Start day used to calculate DST, for example, 0-6 day of the week.
start_mon	INTEGER		Represents the starting month for the timezone, such as 0-11 for the month of the year.
start_pos	INTEGER		Represents the starting position, such as 0 for "First" or 1 for "Last".
sym	nvarchar(30)		Specifies the name of the service type.

Transition_Points Table

Lists the transitions of interest to the Notification System. E.g. Incident_report creation, or reassignment, TT creation TT closure and so on.

- **SQL Name**—notrnr

Field	Data Type	Reference	Remarks
del	INTEGER NOT_NULL		Deleted flag (0=active 1=inactive/marketed as deleted)
enum	INTEGER NOT_NULL		Enumerated value for this entry - specifies ordering in lists and relative values
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
nx_desc	STRING 40		longer description of event
sym	STRING 30 UNIQUE NOT_NULL S_KEY		transition event symbol
tp_use_pri	INTEGER		flag indicating whether priority is meaningful

Task_Type Table

This table contains the list of task types used in the workflow used by Unicenter Service Desk, not those used by CA Workflow.

- **SQL Name**—tskty
- **Object**—tskty

Field	Data Type	Reference	Remarks
code	nvarchar(12)		Primary key of this table.
del	INTEGER	Active_ Boolean_ Table::: enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
description	nvarchar(500)		Identifies the text description of the task.
id	INTEGER		Identifies the numeric ID, which is unique to the table.

Field	Data Type	Reference	Remarks
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Identifies the timestamp for when this record was last modified.
persid	nvarchar(30)		Identifies the Persistent ID: (SystemObjectName:id).
sym	nvarchar(30)		Identifies the name of task.

Type_Of_Contact Table

Reference table to denote the type of issue. examples: complaint, complement and so on.

- **SQL Name**—toc
- **Object**—typecnt

Field	Data Type	Reference	Remarks
del	INTEGER	Active_Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
id	INTEGER		Primary key of this table, this is the unique numeric ID.
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sym	nvarchar(60)		The symbolic value for this Type_Of_Contact.

Urgency Table

List of urgency codes/descriptions used in Unicenter Service Desk applications.

- **SQL Name**—urgnc
- **Object**—urg

Field	Data Type	Reference	Remarks
del	INTEGER	Active_ Boolean_Table: :enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted
enum	INTEGER		Primary key of this table.
id	INTEGER		Identifies the unique (to the table) numeric ID.
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
nx_desc	nvarchar(40)		Provides the description of the urgency level.
sym	nvarchar(12)		Identifies the symbolic name of this urgency.
value	INTEGER		Indicates the numeric representation of this urgency.

User_Query

User scoreboard queries.

- **SQL Name**—usq
- **Object**—usq

Field	Data Type	Reference	Remarks
expanded	INTEGER		
factory	STRING 30		
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
label	STRING 80 NOT_NULL		
last_mod_b_y	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_d_t	LOCAL_TIME		Timestamp of when this record was last modified
obj_persid	STRING 60		Persistent ID (SystemObjectName:id)
parent	INTEGER	usq::id	
persid	STRING 30		Persistent ID (SystemObjectName:id)
query	STRING 30	Cr_Stored_Queries ::code	
query_set	INTEGER		
query_type	INTEGER		
sequence	INTEGER NOT_NULL		

usp_contact Table

This table provides extensions to the ca_contact table that are used only by Unicenter Service Desk products.

- **SQL Name**—usp_contact
- **Object**—cnt

Field	Data Type	Reference	Remarks
c_acctyp_id	INTEGER	Access_Type::id	Identifies the unique (to the table) numeric ID.
c_available	INTEGER		Displays as a check box to indicate that the analyst is available.
c_cm_id1	INTEGER	Contact_Method ::id	Foreign key to the id field of the ct_mth table, this identifies the low priority of the contact method.
c_cm_id2	INTEGER	Contact_Method ::id	Foreign key to the id field of the ct_mth table, this identifies the next level of low priority for the contact method.
c_cm_id3	INTEGER	Contact_Method ::id	Foreign key to the id field of the ct_mth table, this identifies the standard level of priority for the contact method.
c_cm_id4	INTEGER	Contact_Method ::id	Foreign key to the id field of the ct_mth table, this identifies the high level of priority for the contact method.
c_domain	INTEGER	Domain :id	Foreign key to the id field of the dmn table, this is the Data Partition.
c_email_service	nvarchar (30)		(Not used by Service Desk) Identifies the pointer to the access type email service (such as, PROFS, and so on).
c_nx_ref_1	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is a user-defined field.
c_nx_ref_2	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is a user-defined field.

[usp_contact Table](#)

Field	Data Type	Reference	Remarks
c_nx_ref_3	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is a user-defined field.
c_nx_string1	nvarchar (40)		Identifies the emergency Workshift 4 Smag fields.
c_nx_string2	nvarchar (40)		Allows for a user-defined string field.
c_nx_string3	nvarchar (40)		Allows for a user-defined string field.
c_nx_string4	nvarchar (40)		Allows for a user-defined string field.
c_nx_string5	nvarchar (40)		Allows for a user-defined string field.
c_nx_string6	nvarchar (40)		Allows for a user-defined string field.
c_parent	byte(16)		(Not used in Service Desk) Foreign key to the contact_uuid field of the ca_contact table.
c_schedule	nvarchar (30)	Bop_Workshift ::persid	Foreign key to the persistent_id field of the bpwshft table, this is the Analyst's workshift for Auto Assignment.
c_service_type	nvarchar (30)	Service_Desc::code	Foreign key to the code field of the srv_desc table, this identifies the Classic Service Type.
c_timezone	nvarchar (12)	Timezone ::code	Foreign key to the code field of the tz table, this defines the Timezone.
c_val_req	INTEGER		Specifies a Force validation of the userid.
c_vendor	byte(16)	ca_company ::company_uuid	Foreign key to the id filed of the ca_company table, this represents the Vendor.
c_ws_id1	nvarchar (30)	Bop_Workshift ::persid	Foreign key to the persistent_id field of the bpwshft table, this represents a workshift ID.

Field	Data Type	Reference	Remarks
c_ws_id2	nvarchar(30)	Bop_Workshift::persid	Foreign key to the persistent_id field of the bpwshft table, this represents a workshift ID..
c_ws_id3	nvarchar(30)	Bop_Workshift::persid	Foreign key to the persistent_id field of the bpwshft table, this represents a workshift ID.
c_ws_id4	nvarchar(30)	Bop_Workshift::persid	Foreign key to the persistent_id field of the bpwshft table, this represents a workshift ID.
contact_uuid	byte(16)		Primary key of this table.
global_queue_id	INTEGER	Global_Queue_Names::id	Specifies the pointer to the global queue.
ldap_dn	nvarchar(512)		Identifies the ldap dn value for this usp_contact.

usp_organization Table

This table provides extensions to the ca_organization table that are used only by Unicenter Service Desk products.

- **SQL Name**—usp_organization
- **Object**—org

Field	Data Type	Reference	Remarks
iorg_assigned_svr	INTEGER		Deprecated.
iorg_service_type	nvarchar(30)	Service_Desc::code	Foreign key to the code field of the srv_desc table, this is the Classic Service Type.
last_mod	INTEGER		Specifies the timestamp of when this record was last modified.
organization_uuid	byte(16)		Primary key. Unique identifier.

[usp_owned_resource Table](#)

Field	Data Type	Reference	Remarks
owning_contract	INTEGER	Service_Contract::id	Identifies the unique (to the table) numeric ID.

[usp_owned_resource Table](#)

This table provides extensions to the ca_owned_resource table that are used only by Unicenter Service Desk products.

- **SQL Name**—usp_owned_resource
- **Object**—nr

Field	Data Type	Reference	Remarks
nr_argis_id	nvarchar(40)		Identifies the Argis Workflow Asset ID.
nr_bm_rep	INTEGER	busrep id	Foreign key to id field of the busrep table, this is the Change Impact Analyzer repository.
nr_bmlabel	nvarchar(255)		Identifies the Change Impact Analyzer label.
nr_bms	INTEGER	Business_Management_Status::status_no	Foreign key id field of the busstat table, this is the Change Impact Analyzer status.
nr_exp_dt	INTEGER		Identifies the expiration date for the license or lease.
nr_financial_id	nvarchar(40)		Specifies the financial ID number.
nr_nx_ref_1	byte(16)	ca_contact::uuid	Foreign key to the contact_uuid field of the ca_contact table, this is a user-defined contact field.

Field	Data Type	Reference	Remarks
nr_nx_ref_2	byte(16)	ca_contact::uuid	Foreign key to the contact_uuid field of the ca_contact table, this is a user-defined contact field.
nr_nx_ref_3	byte(16)	ca_contact::uuid	Foreign key to the contact_uuid field of the ca_contact table, this is a user-defined contact field.
nr_nx_string1	nvarchar (40)		Specifies the user-defined string field.
nr_nx_string2	nvarchar (40)		Specifies the user-defined string field.
nr_nx_string3	nvarchar (40)		Specifies the user-defined string field.
nr_nx_string4	nvarchar (40)		Specifies the user-defined string field.
nr_nx_string5	nvarchar (40)		Specifies the user-defined string field.
nr_nx_string6	nvarchar (40)		Specifies the user-defined string field.
nr_pr_id	INTEGER	Priority::enum	Enumerated value for this entry, this specifies the ordering in lists and relative values.
nr_prim_skt_id	INTEGER		Specifies the pointer to the primary search key type.
nr_service_type	nvarchar (30)	Service_Desc::code	Indicates a non-editable enum for field.
nr_sla_id	INTEGER		Identifies the sla value for the usp_owned_resource.
nr_wrty_end_dt	INTEGER		Specifies the warranty end date.
nr_wrty_st_dt	INTEGER		Specifies the warranty start date.

Field	Data Type	Reference	Remarks
owned_resource_uuid	byte(16)		Primary key of this table, this is the UUID to uniquely identify the owned resource within tables supporting ownership information.

USP_PREFERENCES Table

Holds information about the Service Desk and Knowledge Tools application preferences.

- **SQL Name**—USP_PREFERENCES
- **Object**—USP_PREFERENCES

Field	Data Type	Reference	Remarks
ANALYST_ID	UUID	ca_contact: :uuid	
ARC_DOCS_TO_DISPLAY	INTEGER		
ASSIGNEE	INTEGER		
AUTHOR	INTEGER		
BU_RESULT	INTEGER		
CLASSIC_RES ULTSET_CONT EXT	INTEGER		Specifies the Classic resultset context menu activation.
CREATED_VIA	INTEGER		
CREATION_DA TE	INTEGER		Specifies the timestamp indicating when this record was created.
CURRENT_ACT ION	INTEGER		
CUSTOM1	INTEGER		
CUSTOM2	INTEGER		
CUSTOM3	INTEGER		
CUSTOM4	INTEGER		

Field	Data Type	Reference	Remarks
CUSTOM5	INTEGER		
CUSTOM_NUM 1	INTEGER		
CUSTOM_NUM 2	INTEGER		
DOC_ID	INTEGER		
DOC_TEMPLAT E	INTEGER		
DOC_TYPE	INTEGER		
DOC_VERSION	INTEGER		
EXPIRATION_D ATE	INTEGER		
GLOBALSD_AC TIVE_ZONE	INTEGER		Specifies the status of the Global Service Desk Active Zone log reader: 0x01 = Reduce popups 0x02 = Close log reader
HITS	INTEGER		
ID	INTEGER NOT_NULL KEY		Specifies the numeric ID that is unique to this table.
INBOX_COUNT ER	INTEGER		
INITIATOR	INTEGER		
ITEM	INTEGER		
KT_REPORT_C ARD_PAST_DA YS	INTEGER		Indicates the Knowledge report card past_days status. This is a user-defined preference.
KT_REPORT_C ARD_SCREEN_ DEFAULT	INTEGER		Indicates the Knowledge report card screen default. this is a user-defined preference.
LAST_ACCEPTE D_DATE	INTEGER		
LAST_MOD_DT	LOCAL_TIME		Specifies the Timestamp for when this record was last modified.

[USP_PREFERENCES Table](#)

Field	Data Type	Reference	Remarks
MODIFY_DATE	INTEGER		
ONE_B_DOC_V	INTEGER		
IEW_MODE			
ONE_B_DOCS_	INTEGER		
TO_DISPLAY			
ONE_B_HIDE_	INTEGER		
DETAILS			
ONE_B_MATCH	INTEGER		
_TYPE			
ONE_B_SEARC	INTEGER		
H_FIELDS			
ONE_B_SEARC	STRING 255		
H_ORDER			
ONE_B_SEARC	INTEGER		
H_TYPE			
ONE_B_WORD	INTEGER		
_PARTS			
OWNER	INTEGER		
PRIMARY_IND	INTEGER		
EX			
PRIORITY	INTEGER		
PRODUCT	INTEGER		
PUBLISHED_D	INTEGER		
ATE			
REVIEW_DATE	INTEGER		
SD_ACCEPTED	INTEGER		
_HITS			
SD_IMPACT	INTEGER		
SD_PRIORITY	INTEGER		
SD_ROOTCAU	INTEGER		
SE			
SD_SEARCH_F	INTEGER		Specifies the Service Desk and Knowledge Tools search fields for requests.
IELDS_CR			

Field	Data Type	Reference	Remarks
SD_SEARCH_F IELDS_ISS	INTEGER		Specifies the Service Desk and Knowledge Tools search fields for issues.
SD_SEVERITY	INTEGER		
SD_URGENCY	INTEGER		
START_DATE	INTEGER		
STATUS	INTEGER		
SUBJECT_EXPE RT	INTEGER		
USER_DEF_ID	INTEGER		
WEB_LAST_LO GIN	LOCAL_TIME		Indicates the time of the last web login.
WEB_POPUP1_ HEIGHT	INTEGER		Specifies the maximum height for pop-up1.
WEB_POPUP1_ WIDTH	INTEGER		Specifies the maximum width for the Web pop-up1.
WEB_POPUP2_ HEIGHT	INTEGER		Specifies the medium Web pop-up2 height.
WEB_POPUP2_ WIDTH	INTEGER		Specifies the Medium Web pop-up2 width.
WEB_POPUP3_ HEIGHT	INTEGER		Specifies the Small Web pop-up3 height.
WEB_POPUP3_ WIDTH	INTEGER		Specifies the Small Web pop-up3 width.
WEB_POPUP4_ HEIGHT	INTEGER		Specifies the extra Small Web pop-up height.
WEB_POPUP4_ WIDTH	INTEGER		Specifies the extra Small Web pop-up4 width.
WEB_PREFERE NCES	INTEGER		Indicates Web Preferences Flags.
WEB_SUPPRES S_TOUR	INTEGER		Specifies the Web suppressor tour flag: 1 = Do not Suppress 2 = Suppress tour page
WEB_TOOLBA R_TAB	INTEGER		Indicates the initial toolbar tab.

[USP_PROPERTIES Table](#)

Field	Data Type	Reference	Remarks
WF_TEMPLATE	INTEGER		

USP_PROPERTIES Table

Provides a list of property/value pairs for Service Desk and Knowledge Tools applications.

- **SQL Name**—USP_PROPERTIES
- **Object**—USP_PROPERTIES

The maximum number of characters (HTML or pure text) allowed in the document's resolution field is 32768 bytes by default. The system Administrator can set this limit based on the type of data being stored. The limit can be set from the Administration tab, Knowledge, Documents, Document Settings.

There is also a built-in limit of 32768 bytes for the document's pure text that will be indexed. If the resolution of a document is larger than the set limit, only the first 32768 bytes of the document will be indexed and the rest will be ignored.

Field	Data Type	Reference	Remarks
ID	INTEGER NOT_NULL KEY		Unique (to the table) Numeric ID
LAST_MOD_DT	LOCAL_TIME		Timestamp of when this record was last modified
PROPERTY_DEFAULT	STRING 32768		
PROPERTY_DESCRIPTION	STRING 255		
PROPERTY_NAME	STRING 100 S_KEY		
PROPERTY_TYPE	STRING 100		
PROPERTY_VALUE	STRING 32768		

Wftpl_Group Table

Used to build the list of Groups that can serve the Task.

- **SQL Name**—wftpl_grp
- **Object**—wftpl_grp

Field	Data Type	Reference	Remarks
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
l_attr	STRING 30		left hand attribute name
l_persid	STRING 60		left hand key
l_sql	INTEGER		left hand sort order
r_attr	STRING 30		right hand attribute name
r_persid	STRING 60		right hand key
r_sql	INTEGER		right hand sort order

Workflow_Task Table

Object workflow tasks. Task is in here because tasks can be added from a change request which means there is no task template to get the task from.

- **SQL Name**—wf
- **Object**—wf

Field	Data Type	Reference	Remarks
actual_duration	INTEGER		Identifies the time spent on the task by the user.
asset	byte(16)	ca_owned_resource::uuid	Foreign key to the id field of the ca_owned_resource table, this is the Asset.

Workflow_Task Table

Field	Data Type	Reference	Remarks
assignee	byte(16)	ca_contact:: uuid	Foreign key to the contact_uuid field of the ca_contact table, this is the Assignee.
completion_date	INTEGER		Specifies the date of the task.
cost	INTEGER		Identifies the cost value for this Workflow task.
creator	byte(16)	ca_contact:: uuid	Foreign key to the contact_uuid field of the ca_contact table, this identifies the person who created the Workflow task.
date_created	INTEGER		Identifies the date and time of when the record was created.
del	INTEGER	Active_Boolean _Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/mark as deleted
description	nvarchar(1000)		Contains the Workflow task notes.
done_by	byte(16)	ca_contact:: uuid	Foreign key to the contact_uuid field of the ca_contact table, this identifies who created the Workflow task.
est_comp_date	INTEGER		Identifies the estimated completion date value for this Workflow task.
est_cost	INTEGER		Identifies the estimated cost value for this Workflow task.

Field	Data Type	Reference	Remarks
est_duration	INTEGER		Specifies the estimated duration value for this Workflow task.
group_id	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is the Group.
group_task	INTEGER		Sets the flag for group_task, as follows: 0=No 1=Yes
id	INTEGER		Primary key of this table.
last_mod_by	byte(16)	ca_contact::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
object_id	INTEGER	chg id	Identifies the id of the object to which this Workflow task belongs.
object_type	nvarchar(30)		Identifies the short name of the object to which this Workflow task belongs.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sequence	INTEGER		Specifies the sequence value for this Workflow task.
start_date	INTEGER		Identifies the date that the status changed to pending.

Workflow_Task_Template Table

Field	Data Type	Reference	Remarks
status	nvarchar(12)	Task_Status::code	Indicates the non-editable key for the status.
support_lev	nvarchar(30)	Service_Desc::code	Foreign key to the code field of the srv_desc table, this is the Classic Service Type.
support_level	nvarchar(30)		Deprecated.
task	nvarchar(12)	Task_Type::code	Foreign key to the code field of the tskty table, this is the Task template.
wf_template	INTEGER	Workflow_Task_Template::id	Foreign key to the id field of the wtpl table, this is the Workflow Task Template.

Workflow_Task_Template Table

Workflow task templates. Information outlined here is used to build the Workflow task.

- **SQL Name**—wtpl
- **Object**—wtpl

Field	Data Type	Reference	Remarks
assignee	byte(16)	ca_contact::uuid	Foreign key to the contact_uuid field of the ca_contact table, this is the Assignee.
auto_assign	INTEGER		Indicates the flag that enables Auto Assignment.
del	INTEGER	Active_Boolean_Table::enum	Specifies the Deleted flag, as follows: 0=Active 1=Inactive/marketed as deleted

Field	Data Type	Reference	Remarks
deleteable	INTEGER		Specifies the Deleteable flag, as follows: 0=N 1=Yes
description	nvarchar(240)		Specifies the textual description.
est_cost	INTEGER		Specifies the estimated cost value for this Workflow_Task_Template.
est_duration	INTEGER		Specifies the estimated duration value for this Workflow_Task_Template.
group_id	byte(16)		Foreign key to the contact_uuid field of the ca_contact table, this is the Group.
id	INTEGER		Primary key of this table
last_mod_by	byte(16)	ca_contact ::uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	INTEGER		Specifies the timestamp of when this record was last modified.
object_attrname	nvarchar(30)		Specifies that this template belongs to the attribute name in the object.
object_attrval	INTEGER		Specifies the object attribute value which drives the template attribute value.
object_type	nvarchar(30)		Identifies the short name of the object.
persid	nvarchar(30)		Persistent ID (SystemObjectName:id).
sequence	INTEGER		Identifies the sequence value for this Workflow_Task_Template.

Field	Data Type	Reference	Remarks
service_type	nvarchar(30)	Service_Desc::code	Foreign key to the code field of the srv_desc table, this is the Classic Service Type.
task	nvarchar(12)	Task_Type::code	Foreign key to the code field of the tskty table, this is the Task template.

wspcol Table

There are one to three rows in this table for every column created or updated by WSP. The columns table_name+column_name+status form a unique key. See also comments in wsp.maj.

- **SQL Name**—wspcol
- **Object**—wspcol

Field	Data Type	Reference	Remarks
description	STRING 300		Description
addl_info	STRING 500		Triggers, QREL stuff, etc.
column_name	STRING 40 NOT_NULL		Column Majic name
dbms_name	STRING 40		DBMS schema name
display_name	STRING 80		Human-readable name
id	INTEGER UNIQUE NOT_NULL KEY		Unique (to the table) Numeric ID
is_cluster	INTEGER		1 = DBMS cluster index
is_descending	INTEGER		1 = DBMS descending index
is_indexed	INTEGER		1 = DBMS-indexed column
is_local	INTEGER		1 = local column
is_not_null	INTEGER		1 = column cannot be null
is_order_by	INTEGER		1 = DBMS order by index
is_required	INTEGER		1 = column required

Field	Data Type	Reference	Remarks
is_skey	INTEGER		1 = column is secondary key
is_unique	INTEGER		1 = DBMS unique index
is_write_new	INTEGER		1 = read-only after creation
is_wsp	INTEGER		1 = table created by WSP
last_mod_by	UUID	ca_contact::uuid	Specifies the UUID of the contact who last modified this record
last_mod_dt	LOCAL_TIME		Timestamp of when this record was last modified
on_ci_set	STRING 80		Value to set on update
on_new_default	STRING 80		Default value of attribute
persid	STRING 30		Persistent ID (SystemObjectName:id)
schema_name	STRING 40		USD schema name
status	INTEGER NOT_NULL		Modification status
string_len	INTEGER		String len
table_name	STRING 40 NOT_NULL		Table Majic name
type	INTEGER		Column type
xrel_table	STRING 80		Operand of SRELBRELQREL

wsptbl Table

There are one to three rows in this table for every table created or updated by WSP. The columns table+name+status form a unique key. See also comments in wsp.maj for interpretation of integer codes.

- **SQL Name**—wsptbl
- **Object**—wsptbl

Field	Data Type	Reference	Remarks
description	STRING 300		Identifies the description of the table.
common_name	STRING 40	e	Specifies the name for the identification column.
dbms_name	STRING 40		Specifies the DBMS schema name.
display_group	STRING 40		Identifies the Grouping to display.
display_name	STRING 80		Specifies the user-readable name.
function_group	STRING 40		Identifies the Security function group.
id	INTEGER UNIQUE NOT_NULL KEY		Specifies the numeric ID, which is also unique to the table.
is_local	INTEGER		Identifies whether the table is local (1 = local table).
is_wsp	INTEGER		Identifies whether the table is created by WSP (1 = table created by WSP).
last_mod_by	UUID	ca_contact: :uuid	Specifies the UUID of the contact who last modified this record.
last_mod_dt	LOCAL_TIMESTAMP		Identifies the Timestamp for when this record was last modified.
methods	STRING 500		Identifies Spell methods.
persid	STRING 30		Identifies the Persistent ID: (SystemObjectName:id)
rel_attr	STRING 40		Specifies the Foreign key column.
schema_name	STRING 40		Specifies the USD schema name.

Field	Data Type	Reference	Remarks
sort_by	STRING 150		Specifies the Default sort sequence.
status	INTEGER NOT_NULL		Identifies the Modification status.
table_name	STRING 40 NOT_NULL		Specifies the Table Majic name.
triggers	STRING 500		Identifies any Triggers for the table.

Appendix B: Objects and Attributes

This appendix lists the objects and attributes that define Unicenter Service Desk. These are needed to build notification text, scoreboard queries, and data partition constraints. The appendix gives a detailed list of the attributes of each object.

The FACTORY Optional statement that accompanies each object (specified in the Note), defines access to the object, including its relation attribute, a common name, the security group that can access it, the type of lists produced, and how those lists can be sorted. If omitted, the object is treated according to default specifications, as described in the Appendix "Object Definition Syntax".

Note: In tables where the DB Field is represented by the term LOCAL, the definition refers to a keyword for specifying that the attribute does not map to a column in a table in a database.

acc_lvls Object

The object details are as follows:

Associated Table: Access_Levels

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: security

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
enum	enum	INTEGER		UNIQUE REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
description	nx_desc	STRING		
persid	persid	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

acctyp Object

The object details are as follows:

Associated Table: Access_Type

Factories: default

REL_ATTR: id COMMON NAME

Common Name: sym

Function Group: security

Attribute	DB Field	Data Type	SREL References	Flags
access_level	access_level	INTEGER	acc_lvls enum	REQUIRED
admin	admin	INTEGER		
call_mgr	call_mgr	INTEGER		
change_mgr	change_mgr	INTEGER		
config	config	INTEGER		
ct_analyst	ct_analyst	INTEGER		
ct_customer	ct_customer	INTEGER		
ct_type	ct_type	INTEGER		
default_flag	default_flag	INTEGER		
delete_flag	del	INTEGER	actbool enum	REQUIRED
description	description	STRING		
domain	domain	INTEGER	dmn id	
override_ctc_domain_flag	domain_flag	INTEGER		REQUIRED
external_auth	external_auth	INTEGER		REQUIRED
form_group	form_group	INTEGER	frmgrp id	
grant_level	grant_level	INTEGER	acc_lvls enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
initial_form	initial_form	STRING		
interface_type	interface_type	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
inventory	inventory	INTEGER		
issue_mgr	issue_mgr	INTEGER		
kcat	kcat	INTEGER		
kd	kd	INTEGER		
kt_admin	kt_admin	INTEGER		
kt_analyst	kt_analyst	INTEGER		
kt_customer	kt_customer	INTEGER		
kt_engineer	kt_engineer	INTEGER		
kt_manager	kt_manager	INTEGER		
kt_type	kt_type	INTEGER		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIM E		
ldap_access_ group	ldap_access_ group	STRING		
notify	notify	INTEGER		
pin_field	pin_field	STRING		
pref_doc	pref_doc	INTEGER		
reference	reference	INTEGER		
sd_admin	sd_admin	INTEGER		
sd_analyst	sd_analyst	INTEGER		
sd_customer	sd_customer	INTEGER		
sd_employee	sd_employee	INTEGER		
security	security	INTEGER		
sym	sym	STRING		REQUIRED S_KEY
user_auth	user_auth	INTEGER		REQUIRED
view_internal	view_internal	INTEGER		
wsp	wsp	INTEGER		

act_type_assoc Object

The object details are as follows:

Associated Table: Act_Type_Assoc

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
act_type	act_type	STRING	act_type code	
code	code	STRING		UNIQUE REQUIRED S_KEY
description	description	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
internal	int1_rsrved	INTEGER		
int2_rsrved	int2_rsrved	INTEGER		
int3_rsrved	int3_rsrved	INTEGER		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
log_me_f	log_me_f	INTEGER		
ob_type	ob_type	STRING		
ob_type_attr	ob_type_attr	STRING		
persid	persid	STRING		
str1_rsrved	str1_rsrved	STRING		
str2_rsrved	str2_rsrved	STRING		
str3_rsrved	str3_rsrved	STRING		
sym	sym	STRING		REQUIRED

actbool Object

The object details are as follows:

Associated Table: Active_Boolean_Table

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER		REQUIRED
description	description	STRING		
enum	enum	INTEGER		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
	last_mod_dt	last_mod_dt	LOCAL_TIME	
sym	sym	STRING		S_KEY

actrbool Object

The object details are as follows:

Associated Table: Active_Reverse_Boolean_Table

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER		REQUIRED
description	description	STRING		
enum	enum	INTEGER		REQUIRED

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_dt	last_mod_dt	LOCAL_TIME		
sym	sym	STRING		S_KEY

ADMIN_TREE Object

The object details are as follows:

Associated Table: admin_tree

Factories: default

REL_ATTR: id

Common Name: CAPTION

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
CAPTION	CAPTION	STRING		
DESCRIPTION	DESCRIPTION	STRING		
HAS_CHILDREN	HAS_CHILDREN	INTEGER		
id	ID	INTEGER		KEY
KT_ADMIN	KT_ADMIN	INTEGER		
KT_KS_CAPTION	KT_KS_CAPTION	STRING		
KT_KS_FLAG	KT_KS_FLAG	INTEGER		
KT_MANAGER	KT_MANAGER	INTEGER		
last_mod_dt	last_mod_dt	LOCAL_TIME		
PARENT_ID	PARENT_ID	INTEGER		
RESOURCE	RESOURCE	STRING		
SD_ADMIN	SD_ADMIN	INTEGER		

alg Object

The object details are as follows:

Associated Table: Act_Log

Factories: default

REL_ATTR: persistent_id

Common Name: description

Function Group: call_mgr

Attribute	DB Field	Data Type	SREL References	Flags
action_desc	action_desc	STRING		
analyst	analyst	UUID	ca_contact uuid	
call_req_id	call_req_id	STRING	call_req persid	
description	description	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
internal	internal	INTEGER		
session	knowledge_	STRING	session	
k_tool	knowledge_t	STRING	ool	
last_mod_dt	last_mod_dt	LOCAL_TIM E		
persid	persid	STRING		
system_tim e	system_tim e	LOCAL_TIM E		
time_spent	time_spent	DURATION		
time_stamp	time_stamp	LOCAL_TIM E		
type	type	STRING	act_type code	

am_asset_map Object

The object details are as follows:

Associated Table: Am_Asset_Map

Factories: default

REL_ATTR: persistent_id

Common Name: dmuuid

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
dmuuid	am_dmuuid	STRING		
domain_id	am_domain_id	INTEGER		
server	am_server	STRING		
type	am_type	INTEGER		
unit_domain_id	am_unit_domain_id	INTEGER		
unit_id	am_unit_id	INTEGER		
id	id	INTEGER		UNIQUE REQUIRED KEY
int1_rsrved	int1_rsrved	INTEGER		
int2_rsrved	int2_rsrved	INTEGER		
ob_persid	ob_persid	STRING		
ob_type	ob_type	STRING		
persid	persid	STRING		
str1_rsrved	str1_rsrved	STRING		
str2_rsrved	str2_rsrved	STRING		
schema_ver	version	INTEGER		REQUIRED

ANI Object

The object details are as follows:

Associated Table: Animator

Factories: default

REL_ATTR: persistent_id

Common Name: name

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
active	a_act	INTEGER		
delta	a_delta	DURATION		
lock	a_lock	STRING		
name	a_name	STRING		S_KEY
org_timer	a_org	LOCAL_TIME		
user_string	a_string	STRING		
timer	a_time	LOCAL_TIME		
id	id	INTEGER		KEY UNIQUE REQUIRED
method	t_method	STRING		S_KEY
target	t_persid	STRING		S_KEY
type	t_type	INTEGER		

arcpur_hist Object

The object details are as follows:

Associated Table: Archive_Purge_History

Factories: default

REL_ATTR: persistent_id

Common Name:

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
chd_obj_deleted	chd_obj_deleted	INTEGER		
delete_flag	del	INTEGER	actbool enum	REQUIRED
end_time	end_time	LOCAL_TIME		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
obj_deleted	obj_deleted	INTEGER		
persid	persid	STRING		
rule_name	rule_name	STRING	arcpur_rule persid	REQUIRED
start_time	start_time	LOCAL_TIME		

arcpur_rule Object

The object details are as follows:

Associated Table: Archive_Purge_Rule

Factories: default

REL_ATTR: persistent_id

Common Name: name

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
add_query	add_query	STRING		
arc_file_name	arc_file_name	STRING		
conf_obj_name	conf_obj_name	STRING		
days_inactive	days_inactive	INTEGER		REQUIRED
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
name	name	STRING		REQUIRED UNIQUE REQUIRED S_KEY
oper_type	oper_type	INTEGER		REQUIRED
persid	persid	STRING		
reoccur_interv	reoccur_interv	DURATION		
sched	sched	STRING	bpwshft persid	

atev Object

The object details are as follows:

Associated Table: Attached_Events

Factories: default

REL_ATTR: persistent_id

Common Name: user_smag

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
cancel_time	cancel_time	LOCAL_TIME		
event_tmpl	event_tmpl	STRING	evt persid	REQUIRED S_KEY
fire_time	fire_time	LOCAL_TIME		
first_fire_time	first_fire_time	LOCAL_TIME		
group_name	group_name	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_dt	last_mod_dt	LOCAL_TIME		
num_fire	num_fire	INTEGER		
obj_id	obj_id	STRING		REQUIRED
owning_ast	owning_ast	INTEGER	attached_sla_id	
persid	persid	STRING		
start_time	start_time	LOCAL_TIME		
status_flag	status_flag	INTEGER		
user_smag	user_smag	STRING		
wait_time	wait_time	DURATION		

atomic_cond Object

The object details are as follows:

Associated Table: Atomic_Condition

Factories: default

REL_ATTR: id

Common Name: description

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
cond_code	cond_code	STRING		
connecter	connecter	INTEGER		REQUIRED
delete_flag	del	INTEGER	actbool enum	REQUIRED
description	description	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
l paran	l paran	INTEGER		REQUIRED
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
lval	lval	STRING	atyp_asc code	REQUIRED
operator	operator	INTEGER		REQUIRED
owning_macr o	owning_macr o	STRING	splmac persid	
persid	persid	STRING		
r paran	r paran	INTEGER		REQUIRED
rval	rval	STRING		
rval_assoc	rval_assoc	STRING	atyp_asc code	
sequence	sequence	INTEGER		REQUIRED

attached_sla Object

The object details are as follows:

Associated Table: Attached_SLA

Factories: default, ttv_slas

REL_ATTR: id

Common Name: ticket_type

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
_mapped_chg	_mapped_chg	INTEGER	chg id	
_mapped_cr	_mapped_cr	STRING	call_req persid	
_mapped_iss	_mapped_iss	STRING	issue persistent_id	
_mapped_iss_wf	_mapped_iss_wf	INTEGER	isswf id	
_mapped_wf	_mapped_wf	INTEGER	wf id	
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
last_ttv_upd	last_ttv_upd	LOCAL_TIME		
map_sdsc	map_sdsc	STRING	srv_desc code	REQUIRED
persid	persid	STRING		
sla_viol_status	sla_viol_status	INTEGER		
ticket_id	ticket_id	INTEGER		REQUIRED S_KEY
ticket_type	ticket_type	STRING		REQUIRED
time_toViolation	time_to_violation	LOCAL_TIME		
ttv_event	ttv_event	STRING	att_evt persid	

attmnt Object

The object details are as follows:

Associated Table: Attachment

Factories: default

REL_ATTR: id

Common Name: created_dt

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
attmnt_name	attmnt_name	STRING		
created_by	created_by	UUID	ca_contact uuid	
created_dt	created_dt	LOCAL_TIME		
delete_flag	del	INTEGER	actbool enum	REQUIRED
description	description	STRING		
exec_cmd	exec_cmd	STRING	rem_ref code	
file_date	file_date	LOCAL_TIME		
file_name	file_name	STRING		
file_size	file_size	INTEGER		
file_type	file_type	STRING		
folder_id	folder_id	INTEGER	attmnt_folder id	
folder_path_ids	folder_path_ids	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
inherit_permission_id	inherit_permission_id	INTEGER		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		

Attribute	DB Field	Data Type	SREL References	Flags
link_only	link_only	INTEGER	bool_tab enum	
orig_file_name	orig_file_name	STRING		
persid	persid	STRING		
read_pgroup	read_pgroup	INTEGER	P_GROUPS id	
rel_file_path	rel_file_path	STRING		
repository	repository	STRING	doc_rep persid	
status	status	STRING		
write_pgroup	write_pgroup	INTEGER	P_GROUPS id	
zip_flag	zip_flag	INTEGER		

attmnt_folder Object

The object details are as follows:

Associated Table: attmnt_folder

Factories: default

REL_ATTR: id

Common Name: folder_name

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
folder_name	folder_name	STRING		
folder_path_ids	folder_path_id_s	STRING		
folder_type	folder_type	INTEGER		
has_children	has_children	INTEGER		
id	id	INTEGER		REQUIRED KEY
inherit_permission_id	inherit_permission_id	INTEGER	attmnt_folder_id	
last_mod_date	last_mod_date	LOCAL_TIME		

Attribute	DB Field	Data Type	SREL References	Flags
parent_id	parent_id	INTEGER	attmnt_folder id	
read_pgroup	read_pgroup	INTEGER	P_GROUPS id	
repository	repository	STRING	doc_rep persid	
write_pgroup	write_pgroup	INTEGER	P_GROUPS id	

attmnt_lrel Object

The object details are as follows:

Associated Table: Attachment_Lrel

Factories: default

REL_ATTR: persistent_id

Common Name: lpid

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		
lpid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

aty Object

The object details are as follows:

Associated Table: Act_Type

Factories: default, chgaty, craty, issaty, mgsaty

REL_ATTR: code

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
chg_default_surve	chg_default_surve	INTEGER	survey_tpl id	
vey	y			
chg_notify_info	chg_notify_info	STRING	splmac persid	
chg_send_survey	chg_send_survey	INTEGER		
y				
chg_survey_method	chg_survey_metho	INTEGER	ct_mth id	
thod	d			
chg_survey_msbody	chg_survey_msgbo	STRING		
gbody	dy			
chg_survey_ms	chg_survey_msigt	STRING		
gttitle	I			
code	code	STRING		UNIQUE REQUIRED S_KEY
cr_default_survey	cr_default_survey	INTEGER	survey_tpl id	
ey				
cr_notify_info	cr_notify_info	STRING	splmac persid	
cr_send_survey	cr_send_survey	INTEGER		
cr_survey_method	cr_survey_method	INTEGER	ct_mth id	
od				
cr_survey_msbody	cr_survey_msbgd	STRING		
ody	y			
cr_survey_msigt	cr_survey_msgtile	STRING		
tle				
delete_flag	del	INTEGER	actbool enum	REQUIRED
description	description	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
cr_flag	flag1	INTEGER	bool_tab enum	
chg_flag	flag2	INTEGER	bool_tab enum	
iss_flag	flag3	INTEGER	bool_tab enum	
mgs_flag	flag4	INTEGER	bool_tab enum	
kd_flag	flag5	INTEGER	bool_tab enum	
id	id	INTEGER		UNIQUE REQUIRED KEY
internal	internal	INTEGER		
iss_default_surv	iss_default_survey	INTEGER	survey_tpl id	
iss_notify_info	iss_notify_info	STRING	splmac persid	
iss_send_survey	iss_send_survey	INTEGER		
iss_survey_met hod	iss_survey_metho d	INTEGER	ct_mth id	
iss_survey_msg body	iss_survey_msdbo dy	STRING		
iss_survey_msg title	iss_survey_msgtitl	STRING		
kd_notify_info	kd_notify_info	STRING	splmac persid	
last_mod_dt	last_mod_dt	LOCAL_TI ME		
mgs_notify_info	mgs_notify_info	STRING	splmac persid	
notify_flag	notify	INTEGER		
notify_msg_ack	notify_ack	STRING		
notify_msg_bod y	notify_body	STRING		
notify_msg_bod y_html	notify_body_html	STRING		
notify_level	notify_level	INTEGER	noturg enum	

Attribute	DB Field	Data Type	SREL References	Flags
notify_msg_title	notify_title	STRING		
persid	persid	STRING		
sym	sym	STRING		REQUIRED S_KEY

audlog Object

The object details are as follows:

Associated Table: Audit_Log

Factories: default

REL_ATTR: code

Common Name: audobj_persid

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
analyst	analyst	UUID	ca_contact uuid	
attr_after_val	attr_after_val	STRING		
attr_before_val	attr_before_val	STRING		
attr_name	attr_name	STRING		
aud_opr	aud_opr	INTEGER		
audobj_name	audobj_name	STRING		
audobj_persid	audobj_persid	STRING		
audobj_trkid	audobj_trkid	STRING		
audobj_uniqueid	audobj_uniqueid	STRING		
change_date	change_date	LOCAL_TIME		
id	id	INTEGER		UNIQUE REQUIRED KEY
int1_rsrved	int1_rsrved	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
int2_rsrved	int2_rsrved	INTEGER		
persid	persid	STRING		
str1_rsrved	str1_rsrved	STRING		

bhvtpl Object

The object details are as follows:

Associated Table: Behavior_Template

Factories: default

REL_ATTR: id

Common Name: description

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
context_attrna me	context_attrnam e	STRING		
context_attrval	context_attrval	INTEGER		
context_type	context_type	STRING		REQUIRED
delete_flag	del	INTEGER	actbool enum	REQUIRED
description	description	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
modified_date	last_mod_dt	LOCAL_TIME		
macro_conditio n	macro_condition	STRING	splmac persid	
object_id	object_id	INTEGER		REQUIRED
object_type	object_type	STRING		REQUIRED
persid	persid	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
transition_errm_sg	transition_errms_g	STRING		
transition_test	transition_test	STRING	splmac_persid	

bmcls Object

The object details are as follows:

Associated Table: Business_Management_Class

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_b_y	last_mod_by	UUID	ca_contact uuid	
last_mod_dt_t	last_mod_dt	LOCAL_TIME		
description	nx_desc	STRING		
persid	persid	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

bmhier Object

The object details are as follows:

Associated Table: Business_Management

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
bm_rep	bm_rep	INTEGER	busrep id	
cost	cost	INTEGER		
child	hier_child	UUID	ca_owned_resource uuid	REQUIRED
parent	hier_parent	UUID	ca_owned_resource uuid	
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
description	nx_desc	STRING		
persid	persid	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

bmlrel Object

The object details are as follows:

Associated Table: Business_Management_Repository_Lrel

Factories: default

REL_ATTR: persistent_id

Common Name: lpid

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		
lpid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

bmrep Object

The object details are as follows:

Associated Table: Business_Management_Repository

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
hostname	hostname	STRING		UNIQUE REQUIRED

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
description	nx_desc	STRING		
password	password	STRING		
persid	persid	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY
userid	userid	STRING		

bms Object

The object details are as follows:

Associated Table: Business_Management_Status

Factories: default

REL_ATTR: status_no

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
description	nx_desc	STRING		
persid	persid	STRING		
status_no	status_no	INTEGER		UNIQUE REQUIRED S_KEY
sym	sym	STRING		UNIQUE REQUIRED S_KEY

bool Object

The object details are as follows:

Associated Table: Boolean_Table

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER		REQUIRED
enum	enum	INTEGER		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
desc	nx_desc	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

BU_TRANS Object

The object details are as follows:

Associated Table: BU_TRANS

Factories: default

REL_ATTR: id

Common Name:

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Document Hit Date	BU_DATE	LOCAL_TIME		
Processed by Bubble Up Flag	BU_PROCESSE D	INTEGER		
Document Rating	BU_RATING	REAL		

Attribute	DB Field	Data Type	SREL References	Flags
Document ID	DOC_ID	INTEGER	SKELETONS id	
No Rating Flag	HIT_NO_VOTE	INTEGER		
Hit Origin	HIT_ORIGIN	INTEGER		
id	ID	INTEGER		REQUIRED KEY
Category Id	INDEX_ID	INTEGER	O_INDEXES id	
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Contact ID	USER_ID	UUID	ca_contact uuid	

ca_cmpny Object

The object details are as follows:

Associated Table: ca_company

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
alias	alias	STRING		
authentication _password	authentication _password	STRING		
authentication _user_name	authentication _user_name	STRING		
bbs	bbs	STRING		
Company Name	company_na me	STRING		
company_type	company_typ e	INTEGER	ca_company_type id	

[ca_cmpny Object](#)

Attribute	DB Field	Data Type	SREL References	Flags
id	company_uui_d	UUID		UNIQUE REQUIRED KEY
creation_date	creation_date	LOCAL_TIME	E	
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME	E	
description	description	STRING		
exclude_registration	exclude_registration	integer		
delete_flag	inactive	integer	actbool enum	
last_update_date	last_update_date	LOCAL_TIME	E	
last_update_user	last_update_user	STRING		
location_uuid	location_uuid	UUID	ca_location location_uuid	
month_fiscal_year_ends	month_fiscal_year_ends	integer		
parent_company_uuid	parent_company_uuid	UUID	ca_company company_uuid	
primary_contact_uuid	primary_contact_uuid	UUID	ca_contact uuid	
version_number	version_number	integer		
web_address	web_address	STRING		

chg Object

The object details are as follows:

Associated Table: Change_Request

Factories: default

REL_ATTR: id

Common Name: chg_ref_num

Function Group: change_mgr

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
actions	actions	STRING		
active	active_flag	INTEGER	bool_tab enum	REQUIRED
actual_comp_date	actual_comp_d	LOCAL_TIME		
cost	actual_cost	INTEGER		
actual_total_time	actual_total_time	DURATION		
affected_contact	affected_contact	UUID	ca_contact uuid	REQUIRED
assignee	assignee	UUID		
backout_plan	backout_plan	STRING		
call_back_date	call_back_date	LOCAL_TIME		
call_back_flag	call_back_flag	INTEGER		
category	category	STRING	chgcate code	
cawf_procid	cawf_procid	STRING		
chg_ref_num	chg_ref_num	STRING		UNIQUE REQUIRED S_KEY
close_date	close_date	LOCAL_TIME		
created_via	created_via	INTEGER	interface id	
effort	effort	STRING		
est_comp_date	est_comp_date	LOCAL_TIME		

Attribute	DB Field	Data Type	SREL References	Flags
est_cost	est_cost	INTEGER		
est_total_time	est_total_time	DURATION		
flag1	flag1	INTEGER		
flag2	flag2	INTEGER		
flag3	flag3	INTEGER		
flag4	flag4	INTEGER		
flag5	flag5	INTEGER		
flag6	flag6	INTEGER		
group	group_id	UUID		
id	id	INTEGER		UNIQUE REQUIRED KEY
impact	impact	INTEGER	impact enum	
justification	justification	STRING		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
modified_date	last_mod_dt	LOCAL_TIME		
log_agent	log_agent	UUID	ca_contact uuid	REQUIRED
macro_predicted_violation	macro_predict_viol	INTEGER		
need_by	need_by	LOCAL_TIME		
open_date	open_date	LOCAL_TIME		
organization	organization	UUID	ca_organization uuid	
parent	parent	INTEGER	chg id	
persistent_id	persid	STRING		
person_contacting	person_contacting	INTEGER	person id	
predicted_sla_violation	predicted_sla_viol	INTEGER		
priority	priority	INTEGER	pri enum	REQUIRED
product	product	INTEGER	product id	

Attribute	DB Field	Data Type	SREL References	Flags
reporting_method	reporting_method	INTEGER	repmethod id	
requestor	requestor	UUID	ca_contact uuid	REQUIRED
resolve_date	resolve_date	LOCAL_TIME		
rootcause	rootcause	INTEGER	rootcause id	
service_date	service_date	LOCAL_TIME		
service_num	service_num	STRING		
slaViolation	slaViolation	INTEGER		
start_date	start_date	LOCAL_TIME		
status	status	STRING	chgstat code	
string1	string1	STRING		
string2	string2	STRING		
string3	string3	STRING		
string4	string4	STRING		
string5	string5	STRING		
string6	string6	STRING		
summary	summary	STRING		
support_lev	support_lev	STRING	srv_desc code	
template_name	template_name	STRING	chg_template template_name	
type_of_contact	type_of_contact	INTEGER	toc id	
user1	user1	STRING		
user2	user2	STRING		
user3	user3	STRING		

chg_tpl Object

The object details are as follows:

Associated Table: Change_Template

Factories: default

REL_ATTR: template_name

Common Name: template_name

Function Group: change_mgr

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
quick_tmpl_type	quick_tmpl_type	INTEGER	quick_tmpl_type enum	REQUIRED
template	template	INTEGER	chg_id	
template_class	template_class	STRING		
template_name	template_name	STRING		UNIQUE REQUIRED S_KEY

chgalg Object

The object details are as follows:

Associated Table: Change_Act_Log

Factories: default

REL_ATTR: id

Common Name: description

Function Group: change_mgr

Attribute	DB Field	Data Type	SREL References	Flags
Description	description	STRING		
action_desc	action_desc	STRING		
analyst	analyst	UUID	ca_contact uuid	
change_id	change_id	INTEGER	chg_id	
id	id	INTEGER		UNIQUE REQUIRED KEY
internal	internal	INTEGER		
knowledge_session	knowledge_session	STRING		
knowledge_tool	knowledge_tool	STRING		
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
system_time	system_time	LOCAL_TIME		
time_spent	time_spent	DURATION		
time_stamp	time_stamp	LOCAL_TIME		
type	type	STRING	act_type code	

chgcat_grp Object

The object details are as follows:

Associated Table: Chgcat_Group

Factories: default

REL_ATTR: persistent_id

Common Name: lpid

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		
lpid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

chgcat Object

The object details are as follows:

Associated Table: Change_Category

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
assignee	assignee	UUID		
auto_assign	auto_assign	INTEGER		
cawf_defid	cawf_defid	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
children_ok	children_ok	INTEGER		REQUIRED
code	code	STRING		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
group	group_id	UUID		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
modified_date	last_mod_dt	LOCAL_TIM E		
organization	organization	UUID	ca_organization uuid	
owning_contr act	owning_cont ract	INTEGER	svc_contract id	
persistent_id	persid	STRING		
schedule	schedule	INTEGER		
service_type	service_type	STRING	srv_desc code	
survey	survey	INTEGER	survey_tpl id	
sym	sym	STRING		REQUIRED S_KEY

chgcat_loc Object

The object details are as follows:

Associated Table: Chgcat_Loc

Factories: default

REL_ATTR: persistent_id

Common Name: Ipid

Function Group:

Attribut e	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		
Ipid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

chgcat_workshift Object

The object details are as follows:

Associated Table: chgcat_workshift

Factories: default

REL_ATTR: persistent_id

Common Name: Ipid

Function Group:

Attribut e	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		

Attribut	DB Field	Data Type	SREL References	Flags
e				
lpid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

chgstat Object

The object details are as follows:

Associated Table: Change_Status

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
active	active	INTEGER		
code	code	STRING		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
hold	hold	INTEGER		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
modified_dat	last_mod_dt	LOCAL_TIME		
e				
persid	persid	STRING		
resolved	resolved	INTEGER		
sym	sym	STRING		REQUIRED

CI_ACTIONS Object

The object details are as follows:

Associated Table: CI_ACTIONS

Factories: default

REL_ATTR: id

Common Name: ACTION_TITLE

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Task Order	ACTION_ORDE R	INTEGER		
Task Title	ACTION_TITLE	STRING		
Contact ID	ANALYST_ID	UUID	ca_contact uuid	
Group ID	GROUP_ID	UUID	ca_contact uuid	
id	ID	INTEGER		REQUIRED KEY
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Predefined	PREDEFINED	INTEGER		
Current Status ID	STATUS_CURR ENT_ID	INTEGER	CI_STATUSES id	
Unpublish Task	UNPUBLISH	INTEGER		
Unretire Task	UNRETIRE	INTEGER		
Workflow Template ID	WF_TEMPLATE _ID	INTEGER	CI_WF_TEMPLATE S id	

CI_ACTIONS_ALTERNATE Object

The object details are as follows:

Associated Table: CI_ACTIONS_ALTERNATE

Factories: default

REL_ATTR: id

Common Name:

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Task ID	ACTION_ID	INTEGER	CI_ACTIONS id	
Contact ID	CONTACT_ID	UUID	ca_contact uuid	
Contact Type	CONTACT_TYPE	INTEGER		
id	ID	INTEGER		REQUIRED KEY
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		

CI_BOOKMARKS Object

The object details are as follows:

Associated Table: CI_BOOKMARKS

Factories: default

REL_ATTR: id

Common Name: BOOKMARK_TITLE

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Bookmark Title	BOOKMARK_TITL E	STRING		
Document ID	DOCUMENT_ID	INTEGER	SKELETONS id	
id	ID	INTEGER		REQUIRED KEY

Attribute	DB Field	Data Type	SREL References	Flags
Last Modified Date	LAST_MOD_DT	LOCAL_TIM E		
Contact ID	USER_ID	UUID	ca_contact	uuid

CI_DOC_LINKS Object

The object details are as follows:

Associated Table: CI_DOC_LINKS

Factories: default

REL_ATTR: id

Common Name:

Function Group:

Attributet	DB Field	Data Type	SREL References	Flags
Document ID 1	DOC_ID1	INTEGER	SKELETONS id	
Document ID 2	DOC_ID2	INTEGER	SKELETONS id	
id	ID	INTEGER		REQUIRED KEY
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		

CI_DOC_TEMPLATES Object

The object details are as follows:

Associated Table: CI_DOC_TEMPLATES

Factories: default

REL_ATTR: id

Common Name: TEMPLATE_NAME

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		REQUIRED KEY
Default Template Flag	IS_DEFAULT	INTEGER		
Predefined Flag	IS_PREDEFINED	INTEGER		
Last Modified Date	LAST_MOD_DT	LOCAL_TIM E		
Template HTML	PAGE_HTML	STRING		
Template Name	TEMPLATE_NAME	STRING		

CI_DOC_TYPES Object

The object details are as follows:

Associated Table: CI_DOC_TYPES

Factories: default

REL_ATTR: id

Common Name: DOC_TYPE_TXT

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Document Type	DOC_TYPE_TXT	STRING		
id	ID	INTEGER		KEY

Attribute	DB Field	Data Type	SREL References	Flags
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		

CI_PRIORITIES Object

The object details are as follows:

Associated Table: CI_PRIORITIES

Factories: default

REL_ATTR: id

Common Name: PRIORITY

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		KEY
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Workflow Priority	PRIORITY	STRING		

CI_STATUSES Object

The object details are as follows:

Associated Table: CI_STATUSES

Factories: default

REL_ATTR: id

Common Name: STATUS

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		KEY
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Predefined Flag	PREDEFINED	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
Status Name	STATUS	STRING		
Status Description	STATUS_DESCRIPTION	STRING		
Status Order	STATUS_ORDER	INTEGER		

CI_WF_TEMPLATES Object

The object details are as follows:

Associated Table: CI_WF_TEMPLATES

Factories: default

REL_ATTR: id

Common Name: WF_NAME

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		KEY
Default Flag	IS_DEFAULT	INTEGER		
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Workflow Template Description	WF_DESCRIPTION	STRING		
Workflow Template Name	WF_NAME	STRING		

cmth Object

The object details are as follows:

Associated Table: Contact_Method

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
cm_template	cm_template	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
description	nx_desc	STRING		
persid	persid	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY
write_file	write_file	INTEGER		

cnote Object

The object details are as follows:

Associated Table: Note_Board

Factories: default

REL_ATTR: persistent_id

Common Name: text

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
active	active_flag	INTEGER	bool_tab enum	REQUIRED
close_date	close_date	LOCAL_TIME		
cnote_type	cnote_type	INTEGER		
control_grou	control_grou	UUID		
p	p			
del	del	INTEGER		
id	id	INTEGER		UNIQUE REQUIRED KEY
internal	internal	INTEGER		
location	loc_id	UUID	ca_location location_uuid	
organization	organization	UUID	ca_organization uuid	
persid	persid	STRING		
posted_by	posted_by	UUID	ca_contact uuid	
posted_date	posted_date	LOCAL_TIME		
text	text	STRING		

cnt Object

The object details are as follows:

Associated Table: ca_contact, usp_contact

Factories: default, agt, cst, grp

REL_ATTR: id

Common Name: combo_name

Function Group: inventory

ca_contact Table

Attribute	DB Field	Data Type	SREL References	Flags
admin_org	admin_organization_uuid	UUID	ca_organization_uuid	
alias	alias	STRING		
alt_phone	alt_phone_number	STRING		
contact_num	alternate_identifier	STRING		
notes	comment	STRING		
company	company_uuid	UUID	ca_company_uuid	
type	contact_type	integer	ca_contact_type_id	
id	contact_uuid	UUID		UNIQUE REQUIRED KEY
billing_code	cost_center	INTEGER	ca_resource_cost_center_id	
creation_date	creation_date	LOCAL_TIME		
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		
dept	department	INTEGER	ca_resource_department_id	
email_address	email_address	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
exclude_regi stration	exclude_regs tration	integer		
fax_phone	fax_number	STRING		
first_name	first_name	STRING		
floor_locatio n	floor_location	STRING		
delete_flag	inactive	integer	actbool enum	
job_function	job_function	integer		
position	job_title	integer	ca_job_title id	
last_name	last_name	STRING		
last_mod	last_update_d ate	LOCAL_TIME		
last_mod_by	last_update_u ser	STRING		
location	location_uuid	UUID	ca_location location_uuid	
mail_stop	mail_stop	STRING		
middle_nam e	middle_name	STRING		
mobile_phon e	mobile_phone _number	STRING		
organization	organization_ uuid	UUID	ca组织 n uuid	
pemail_addr ess	pager_email_ address	STRING		
beeper_phon e	pager_numbe r	STRING		
phone numb er	pri_phone_nu mber	STRING		
room_locatio n	room_location	STRING		
supervisor_c ontact_uuid	supervisor_co ntact_uuid	UUID	ca_contact uuid	
userid	userid	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
version_num	version_numb	integer		
ber	er			
usp_contact Table				
Attribute	DB Field	Data Type	SREL References	Flags
access_type	c_acctyp_id	INTEGER	acctyp id	
available	c_available	INTEGER		
notify_metho	c_cm_id1	INTEGER	ct_mth id	
d1				
notify_metho	c_cm_id2	INTEGER	ct_mth id	
d2				
notify_metho	c_cm_id3	INTEGER	ct_mth id	
d3				
notify_metho	c_cm_id4	INTEGER	ct_mth id	
d4				
domain	c_domain	INTEGER	dmn id	
c_email_serv	c_email_servi	STRING		
ice	ce			
c_nx_ref_1	c_nx_ref_1	UUID		
c_nx_ref_2	c_nx_ref_2	UUID		
c_nx_ref_3	c_nx_ref_3	UUID		
c_nx_string1	c_nx_string1	STRING		
c_nx_string2	c_nx_string2	STRING		
c_nx_string3	c_nx_string3	STRING		
c_nx_string4	c_nx_string4	STRING		
c_nx_string5	c_nx_string5	STRING		
c_nx_string6	c_nx_string6	STRING		
c_parent	c_parent	UUID		
schedule	c_schedule	STRING	bpwshft persid	
service_type	c_service_typ	STRING	srv_desc code	e

Attribute	DB Field	Data Type	SREL References	Flags
timezone	c_timezone	STRING	tz code	
c_val_req	c_val_req	INTEGER		
vendor	c_vendor	UUID	ca_company company_uui d	
notify_ws1	c_ws_id1	STRING	bpwshft persid	
notify_ws2	c_ws_id2	STRING	bpwshft persid	
notify_ws3	c_ws_id3	STRING	bpwshft persid	
notify_ws4	c_ws_id4	STRING	bpwshft persid	
id	contact_uuid	UUID		UNIQUE REQUIRED KEY
global_queue_id	global_queue_id	INTEGER	g_queue_name es id	
ldap_dn	ldap_dn	STRING		
ldap_dn	ldap_dn	STRING		

cost_cntr Object

The object details are as follows:

Associated Table: ca_resource_cost_center

Factories: default

REL_ATTR: id

Common Name: name

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
creation_date	creation_date	LOCAL_TIME		

Attribute	DB Field	Data Type	SREL References	Flags
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		
exclude_registration	exclude_registration	integer		
id	id	INTEGER		UNIQUE REQUIRED KEY
delete_flag	inactive	integer	actbool enum	
last_mod	last_update_date	LOCAL_TIME		
last_update_user	last_update_user	STRING		
name	name	STRING		
version_number	version_number	integer		

country Object

The object details are as follows:

Associated Table: ca_country

Factories: default

REL_ATTR: id

Common Name: name

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
creation_date	creation_date	LOCAL_TIME		
creation_user	creation_user	STRING		
description	description	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
delete_flag	inactive	integer	actbool enum	

Attribute	DB Field	Data Type	SREL References	Flags
last_mod	last_update_date	LOCAL_TIME		
last_update_use	last_update_user	STRING		
r				
name	name	STRING		
version_number	version_number	integer		

cr Object

The object details are as follows:

Associated Table: Call_Req

Factories: default

REL_ATTR: persistent_id

Common Name: ref_num

Function Group: call_mgr

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
active	active_flag	INTEGER	bool_tab enum	REQUIRED
affected_resource	affected_rc	UUID	ca_owned_resource uuid	
assignee	assignee	UUID		
call_back_date	call_back_date	LOCAL_TIME		
		E		
call_back_flag	call_back_flag	INTEGER		
category	category	STRING	prob_ctg persid	
change	change	INTEGER	chg id	
charge_back_id	charge_back_id	STRING		
		d		
close_date	close_date	LOCAL_TIME		
		E		
cr_ticket	cr_ticket	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
created_via	created_via	INTEGER	interface id	
customer	customer	UUID	ca_contact uuid	REQUIRED
event_token	event_token	STRING		
extern_ref	extern_ref	STRING		
group	group_id	UUID		
id	id	INTEGER		UNIQUE REQUIRED KEY
impact	impact	INTEGER	impact enum	
incident_priority	incident_priority	INTEGER		
last_act_id	last_act_id	STRING		
last_mod_dt	last_mod_dt	LOCAL_TIM E		
log_agent	log_agent	UUID	ca_contact uuid	REQUIRED
macro_predicted_violation	macro_predict _viol	INTEGER		
open_date	open_date	LOCAL_TIM E		
parent	parent	STRING	call_req persid	
persistent_id	persid	STRING		
predicted_sla_violation	predicted_sla_ viol	INTEGER		
priority	priority	INTEGER	pri enum	REQUIRED
problem	problem	STRING		
ref_num	ref_num	STRING		UNIQUE REQUIRED S_KEY
resolve_date	resolve_date	LOCAL_TIM E		
rootcause	rootcause	INTEGER	rootcause id	
extern_token	sched_token	STRING		
severity	severity	INTEGER	sevrty enum	

Attribute	DB Field	Data Type	SREL References	Flags
slaViolation	slaViolation	INTEGER		
baseTemplate	solution	STRING	call_req persid	
status	status	STRING	cr_stat code	
string1	string1	STRING		
string2	string2	STRING		
string3	string3	STRING		
string4	string4	STRING		
string5	string5	STRING		
string6	string6	STRING		
summary	summary	STRING		
support_lev	support_lev	STRING	srv_desc code	
template_name	template_name	STRING	cr_template template_name	
time_spent_su_m	time_spent_su_m	DURATION		
type	type	STRING	crt code	
urgency	urgency	INTEGER	urgncy enum	

cr_prp Object

The object details are as follows:

Associated Table: Req_Property

Factories: default

REL_ATTR: id

Common Name: label

Function Group: call_mgr

Attribute	DB Field	Data Type	SREL References	Flags
description	description	nvarchar (240)		
id	id	INTEGER		Primary key of this table.

Attribute	DB Field	Data Type	SREL References	Flags
label	label	nvarchar (80)		REQUIRED
last_mod_by	last_mod_by	byte(16)	ca_contact uuid	
last_mod_dt	last_mod_dt	INTEGER		
owning_cr	owning_cr	nvarchar (30)	call_req persid	REQUIRED
persid	persid	STRING		
required	required	INTEGER		REQUIRED
sample	sample	nvarchar (240)		
sequence	sequence	INTEGER		REQUIRED
value	value	nvarchar (240)		

cr_prptpl Object

The object details are as follows:

Associated Table: Req_Property_Template

Factories: default

REL_ATTR: id

Common Name: label

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
code	code	STRING		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
label	label	STRING		REQUIRED
last_mod_by	last_mod_by	UUID	ca_contact uuid	

Attribute	DB Field	Data Type	SREL References	Flags
last_mod_dt	last_mod_dt	LOCAL_TIME		
owning_area	owning_area	STRING	prob_ctg persid	REQUIRED
persid	persid	STRING		
required	required	INTEGER		REQUIRED
sample	sample	STRING		
sequence	sequence	INTEGER		REQUIRED

cr_tpl Object

The object details are as follows:

Associated Table: Cr_Template

Factories: default

REL_ATTR: template_name

Common Name: template_name

Function Group: call_mgr

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
quick_tmpl_type	quick_tmpl_type	INTEGER	quick_tpl_types enum	REQUIRED
template	template	STRING	call_req persid	
template_class	template_class	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
template_name	template_name	STRING	e	UNIQUE REQUIRED S_KEY

crs Object

The object details are as follows:

Associated Table: Cr_Status

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
active	active	INTEGER		
code	code	STRING		UNIQUE REQUIRED S_KEY
cr_flag	cr_flag	INTEGER		
delete_flag	del	INTEGER	actbool enum	REQUIRED
hold	hold	INTEGER		
id	id	INTEGER		UNIQUE REQUIRED KEY
in_flag	in_flag	INTEGER		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
pr_flag	pr_flag	INTEGER		
resolved	resolved	INTEGER		
sym	sym	STRING		REQUIRED

crsol Object

The object details are as follows:

Associated Table: Call_Solution

Factories: default

REL_ATTR: persistent_id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
cr_count	cr_count	INTEGER		
cr_flag	cr_flag	INTEGER		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
in_flag	in_flag	INTEGER		
last_mod_dt	last_mod_dt	LOCAL_TIME		
desc	nx_desc	STRING		
persid	persid	STRING		
pr_flag	pr_flag	INTEGER		
solution_approve_d	sapproved	INTEGER	bool_tab enum	
solution_name	sname	STRING		
solution	solution	STRING		
sym	sym	STRING		REQUIRED S_KEY
tcode	tcode	INTEGER		

crsq Object

The object details are as follows:

Associated Table: Cr_Stored_Queries

Factories: default, sqchg, sqcr, sqiss

REL_ATTR: code

Common Name: description

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
code	code	STRING		UNIQUE REQUIRED S_KEY
count_url	count_url	STRING		
where_clause	criteria	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
label	label	STRING		
last_mod_by	last_mod_b y	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIM E		
obj_type	obj_type	STRING		
persid	persid	STRING		

crt Object

The object details are as follows:

Associated Table: Call_Req_Type

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
code	code	nvarchar (10)		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
display_name	display_name	nvarchar (30)		
id	id	INTEGER		UNIQUE REQUIRED KEY
description	nx_desc	nvarchar (30)		
persid	persid	nvarchar (30)		
sym	sym	nvarchar (30)		REQUIRED

ctab Object

The object details are as follows:

Associated Table: Controlled_Table

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
desc	nx_desc	nvarchar (40)		
obj_name	obj_name	nvarchar (30)		
persid	persid	nvarchar (30)		
sym	sym	nvarchar (30)		UNIQUE REQUIRED S_KEY

ctimer Object

The object details are as follows:

Associated Table: Cr_Call_Timers

Factories: default

REL_ATTR: persistent_id

Common Name: color

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
beep	beep	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
color	color	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
text	text	STRING		
threshold	threshold	DURATION		REQUIRED

ctp Object

The object details are as follows:

Associated Table: ca_contact_type

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
creation_date	creation_date	LOCAL_TIME		
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		
description	description	STRING		
exclude_registration	exclude_registration	integer		
id	id	INTEGER		UNIQUE REQUIRED KEY
delete_flag	inactive	integer	actbool enum	

Attribute	DB Field	Data Type	SREL References	Flags
last_mod_dt	last_update_date	LOCAL_TIME		
last_mod_by	last_update_user	STRING		
sym	name	STRING		
user_uuid	user_uuid	UUID	ca_contact_uuid	
version_number	version_number	integer		
view_internal	view_internal	integer		

dcon Object

The object details are as follows:

Associated Table: Domain_Constraint

Factories: default

REL_ATTR: id

Common Name: constraint_majic

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
enum	enum	INTEGER		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
description	nx_desc	STRING		
persid	persid	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

dcon_typ Object

The object details are as follows:

Associated Table: Domain_Constraint_Type

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
enum	enum	INTEGER		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
description	nx_desc	nvarchar (40)		
persid	persid	nvarchar (30)		
sym	sym	nvarchar (12)		UNIQUE REQUIRED S_KEY

dept Object

The object details are as follows:

Associated Table: ca_resource_department

Factories: default

REL_ATTR: id

Common Name: name

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
creation_date	creation_date	LOCAL_TIME		

Attribute	DB Field	Data Type	SREL References	Flags
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		
exclude_registration	exclude_registration	integer		
id	id	INTEGER		UNIQUE REQUIRED KEY
delete_flag	inactive	integer	actbool enum	
last_mod	last_update_date	LOCAL_TIME		
last_update_user	last_update_user	STRING		
name	name	STRING		
version_number	version_number	integer		

dlgsrvr Object

The object details are as follows:

Associated Table: Delegation_Server

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
anon_userid	anon_userid	STRING		
appl_addr	appl_addr	STRING		
default_assignee	default_assignee	UUID	ca_contact uuid	
default_userid	default_userid	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
metafile	metafile	STRING		
description	nx_desc	STRING		
password	password	STRING		
server	server	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY
transport	transport	INTEGER		

dmn Object

The object details are as follows:

Associated Table: Domain

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod	last_mod_dt	LOCAL_TIME		
desc	nx_desc	STRING		
persid	persid	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

doc_rep Object

The object details are as follows:

Associated Table: Document_Repository

Factories: default

REL_ATTR: persistent_id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
archive_path	archive_path	STRING		
archive_type	archive_type	INTEGER		
cgi_path	cgi_path	STRING		
default_rep	default_rep	INTEGER		
delete_flag	del	INTEGER	actbool enum	REQUIRED
file_limit_size	file_limit_size	INTEGER		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
prohibited_file	prohibited_ext	STRING		
protocol	protocol	STRING		
repository_type	repository_type	INTEGER		
e	e			
retrieve_path	retrieve_path	STRING		
server	server	STRING		
servlet_path	servlet_path	STRING		
sym	sym	STRING		REQUIRED S_KEY

Attribute	DB Field	Data Type	SREL References	Flags
upload_path	upload_path	STRING		

EBR_ACRONYMS Object

The object details are as follows:

Associated Table: EBR_ACRONYMS

Factories: default

REL_ATTR: id

Common Name: ACRONYM

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Acronym	ACRONYM	STRING		
id	ID	INTEGER		REQUIRED KEY
Last Modified	LAST_MOD_DT	LOCAL_TIME		
Date				

EBR_FULLTEXT Object

The object details are as follows:

Associated Table: EBR_FULLTEXT

Factories: default

REL_ATTR: id

Common Name: FULL_WORD

Function Group:

Attribute	DB Field	Data Type	SREL Reference	Flags
			s	
DOC_TYPE	DOC_TYPE	INTEGER		
ENTITY_ID	ENTITY_ID	INTEGER		

Attribute	DB Field	Data Type	SREL Reference	Flags
			s	
FULL_WORD	FULL_WORD	STRING		
FULL_WORD_REVERSE	FULL_WORD_REVERSE	STRING		
id	ID	INTEGER		REQUIRED KEY
PERMISSION_INDEX_ID	PERMISSION_INDE_X_ID	INTEGER		
PRODUCT	PRODUCT	STRING		
SHORT_WORD	SHORT_WORD	STRING		
TABLE_ID	TABLE_ID	INTEGER		
WORD_COUNT	WORD_COUNT	INTEGER		
WORD_COUNT_PROBLEM	WORD_COUNT_PROBLEM	INTEGER		
WORD_COUNT_RESOLUTION	WORD_COUNT_RESOLUTION	INTEGER		
WORD_COUNT_SUMMARY	WORD_COUNT_SUMMARY	INTEGER		
WORD_COUNT_TITLE	WORD_COUNT_TITLE	INTEGER		
WORD_IDF	WORD_IDF	INTEGER		
WORD_ORDER	WORD_ORDER	INTEGER		
WORD_TYPE	WORD_TYPE	INTEGER		

EBR_FULLTEXT_ADMIN Object

The object details are as follows:

Associated Table: EBR_FULLTEXT_ADMIN

Factories: default

REL_ATTR: id

Common Name: FULL_WORD

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
DOC_TYPE	DOC_TYPE	INTEGER		
ENTITY_ID	ENTITY_ID	INTEGER		
FULL_WORD	FULL_WORD	STRING		
FULL_WORD_REVERSE	FULL_WORD_REV	STRING		
VERSE	RSE			
id	ID	INTEGER		REQUIRED KEY
PERMISSION_INDEX_ID	PERMISSION_INDEX_ID	INTEGER		
PRODUCT	PRODUCT	STRING		
SHORT_WORD	SHORT_WORD	STRING		
TABLE_ID	TABLE_ID	INTEGER		
WORD_COUNT	WORD_COUNT	INTEGER		
WORD_COUNT_PROBLEM	WORD_COUNT_PR	INTEGER		
WORD_COUNT_RESOLUTION	WORD_COUNT_RE	INTEGER		
WORD_COUNT_SUMMARY	WORD_COUNT_SU	INTEGER		
WORD_COUNT_TITLE	WORD_COUNT_TIT	INTEGER		
WORD_IDF	WORD_IDF	INTEGER		
WORD_ORDER	WORD_ORDER	INTEGER		
WORD_TYPE	WORD_TYPE	INTEGER		

EBR_FULLTEXT_SD Object

The object details are as follows:

Associated Table: EBR_FULLTEXT_SD

Factories: default

REL_ATTR: id

Common Name: FULL_WORD

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
DOC_TYPE	DOC_TYPE	INTEGER		
ENTITY_ID	ENTITY_ID	INTEGER		
FULL_WORD	FULL_WORD	STRING		
FULL_WORD_R VERSE	FULL_WORD_R EVERSE	STRING		
id	ID	INTEGER		REQUIRED KEY
PERMISSION_IN DEX_ID	PERMISSION_I NDEX_ID	INTEGER		
PRODUCT	PRODUCT	STRING		
SHORT_WORD	SHORT_WORD	STRING		
TABLE_ID	TABLE_ID	INTEGER		
WORD_COUNT	WORD_COUNT	INTEGER		
WORD_COUNT_P ROBLEM	WORD_COUNT_P _PROBLEM	INTEGER		
WORD_COUNT_R ESOLUTION	WORD_COUNT_R _RESOLUTION	INTEGER		
WORD_COUNT_S UMMARY	WORD_COUNT_S _SUMMARY	INTEGER		
WORD_COUNT_T ITLE	WORD_COUNT_T _TITLE	INTEGER		
WORD_IDF	WORD_IDF	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
WORD_ORDER	WORD_ORDER	INTEGER		
WORD_TYPE	WORD_TYPE	INTEGER		

EBR_FULLTEXT_SD_ADM Object

The object details are as follows:

Associated Table: EBR_FULLTEXT_SD_ADM

Factories: default

REL_ATTR: id

Common Name: FULL_WORD

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
DOC_TYPE	DOC_TYPE	INTEGER		
ENTITY_ID	ENTITY_ID	INTEGER		
FULL_WORD	FULL_WORD	STRING		
FULL_WORD_REV VERSE	FULL_WORD_REV VERSE	STRING		
id	ID	INTEGER		REQUIRED KEY
PERMISSION_IN DEX_ID	PERMISSION_IND EX_ID	INTEGER		
PRODUCT	PRODUCT	STRING		
SHORT_WORD	SHORT_WORD	STRING		
TABLE_ID	TABLE_ID	INTEGER		
WORD_COUNT	WORD_COUNT	INTEGER		
WORD_COUNT_PR BLEM	WORD_COUNT_PR BLEM	INTEGER		
WORD_COUNT_RE SOLUTION	WORD_COUNT_RE SOLUTION	INTEGER		
WORD_COUNT_S UMMARY	WORD_COUNT_S UMMARY	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
WORD_COUNT_TITLE	WORD_COUNT_TI	INTEGER		
WORD_IDF	WORD_IDF	INTEGER		
WORD_ORDER	WORD_ORDER	INTEGER		
WORD_TYPE	WORD_TYPE	INTEGER		

EBR_INDEXING_QUEUE Object

The object details are as follows:

Associated Table: EBR_INDEXING_QUEUE

Factories: default

REL_ATTR: id

Common Name: OBJ_PERSID

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
ACTION	ACTION	INTEGER		
ACTION_DATE	ACTION_DATE	DATE		
id	ID	INTEGER		REQUIRED KEY
INDEXED	INDEXED	INTEGER		
OBJ_PERSID	OBJ_PERSID	STRING		
PRIORITY	PRIORITY	INTEGER		
TEXT	TEXT	STRING		

EBR_KEYWORDS Object

The object details are as follows:

Associated Table: EBR_KEYWORDS

Factories: default

REL_ATTR: id

Common Name: FULL_WORD

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
ENTITY_ID	ENTITY_ID	INTEGER		
EXT_TABLE_ID	EXT_TABLE_ID	INTEGER		
FULL_WORD	FULL_WORD	STRING		
id	ID	INTEGER		REQUIRED KEY

EBR_LOG Object

The object details are as follows:

Associated Table: EBR_LOG

Factories: default

REL_ATTR: id

Common Name: SEARCH_TEXT

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
ASKED_DATE	ASKED_DATE	LOCAL_TIME		
BEST_IDS	BEST_IDS	STRING		
EXTERNAL_ID	EXTERNAL_ID	STRING		
FILTER_DATA	FILTER_DATA	STRING		
FUZZINESS	FUZZINESS	INTEGER		
id	ID	INTEGER		KEY

Attribute	DB Field	Data Type	SREL Reference s	Flags
KEYWORDS	KEYWORDS	STRING		
MATCH_TYPE	MATCH_TYPE	INTEGER		
METHOD_PERFOR MANCE	METHOD_PERFORMA NCE	INTEGER		
METHOD_TYPE	METHOD_TYPE	INTEGER		
NUM_MATCHES	NUM_MATCHES	INTEGER		
ORDER_DIRECTIO N	ORDER_DIRECTION	INTEGER		
PRIMARY_ORDER	PRIMARY_ORDER	STRING		
ROWS_FOUND	ROWS_FOUND	INTEGER		
ROWS_TO_FETCH	ROWS_TO_FETCH	INTEGER		
SEARCH_IN	SEARCH_IN	INTEGER		
SEARCH_QUALITY	SEARCH_QUALITY	INTEGER		
SEARCH_TEXT	SEARCH_TEXT	STRING		
SEARCH_TYPE	SEARCH_TYPE	INTEGER		
SECONDARY_ORDE R	SECONDARY_ORDER	INTEGER		
SESSION_ID	SESSION_ID	INTEGER		
SQL_TEXT	SQL_TEXT	STRING		
TOP_MATCH_ID	TOP_MATCH_ID	INTEGER		
UNIQUE_WORD_C OUNT	UNIQUE_WORD_CO UNT	INTEGER		
USER_ID	USER_ID	STRING		
WORD_COUNT	WORD_COUNT	INTEGER		

EBR_METRICS Object

The object details are as follows:

Associated Table: EBR_METRICS

Factories: default

REL_ATTR: id

Common Name: METRIC

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Comments	COMMENTS	STRING		
id	ID	INTEGER		REQUIRED KEY
Metric	METRIC	STRING		
Weight	WEIGHT	REAL		

EBR_NOISE_WORDS Object

The object details are as follows:

Associated Table: EBR_NOISE_WORDS

Factories: default

REL_ATTR: id

Common Name: NOISE_WORD

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		REQUIRED KEY
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Noise Word	NOISE_WORD	STRING		

EBR_PATTERNS Object

The object details are as follows:

Associated Table: EBR_PATTERNS

Factories: default

REL_ATTR: id

Common Name: PATTERN_NAME

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		REQUIRED KEY
PATTERN_DEFAL	PATTERN_DEFAL	STRING		
LT	T			
PATTERN_NAME	PATTERN_NAME	STRING		
PATTERN_VALUE	PATTERN_VALUE	STRING		
PATTERN_VALUE_	PATTERN_VALUE_	STRING		
_ADM	ADM			

EBR_PREFIXES Object

The object details are as follows:

Associated Table: EBR_PREFIXES

Factories: default

REL_ATTR: id

Common Name: PREFIX

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		REQUIRED KEY
Prefix	PREFIX	STRING		

EBR_PROPERTIES Object

The object details are as follows:

Associated Table: EBR_PROPERTIES

Factories: default

REL_ATTR: id

Common Name: PROPERTY_NAME

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		REQUIRED KEY
PROPERTY_ADMIN	PROPERTY_ADMIN	INTEGER		
	N			
PROPERTY_DEFAULT	PROPERTY_DEFAULT	STRING		
	ULT			
PROPERTY_NAME	PROPERTY_NAME	STRING		S_KEY
	E			
PROPERTY_TYPE	PROPERTY_TYPE	STRING		
PROPERTY_VALUE	PROPERTY_VALUE	STRING		
	E			
PROPERTY_VALUE_A	PROPERTY_VALUE_A	STRING		
PROPERTY_VALUE_DM	DM			

EBR_SUBSTITS Object

The object details are as follows:

Associated Table: EBR_SUBSTITS

Factories: default

REL_ATTR: id

Common Name: SYMBOL1

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		REQUIRED KEY
SYMBOL1	SYMBOL1	STRING		
SYMBOL2	SYMBOL2	STRING		

EBR_SUFFIXES Object

The object details are as follows:

Associated Table: EBR_SUFFIXES

Factories: default

REL_ATTR: id

Common Name: SUFFIX

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		REQUIRED KEY
Suffix	SUFFIX	STRING		

EBR_SYNONYMS Object

The object details are as follows:

Associated Table: EBR_SYNONYMS

Factories: default

REL_ATTR: id

Common Name: KEYWORD1

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		REQUIRED KEY
Keyword 1	KEYWORD1	STRING		
Keyword 2	KEYWORD2	STRING		
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		

EBR_SYNONYMS_ADMIN Object

The object details are as follows:

Associated Table: EBR_SYNONYMS_ADMIN

Factories: default

REL_ATTR: id

Common Name: KEYWORD1

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		REQUIRED KEY
Keyword 1	KEYWORD1	STRING		
Keyword 2	KEYWORD2	STRING		
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		

ES_CONSTANTS Object

The object details are as follows:

Associated Table: ES_CONSTANTS

Factories: default

REL_ATTR: id

Common Name: NAME

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Comments	COMMENTS	STRING		
id	ID	INTEGER		REQUIRED KEY
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Name	NAME	STRING		
Property ID	PROPERTYID	INTEGER		
Property Value	PROPVVALUE	INTEGER		

ES_NODES Object

The object details are as follows:

Associated Table: ES_NODES

Factories: default

REL_ATTR: id

Common Name: NODE_SHORT_DESC

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
DISPLAYED_T EXT	DISPLAYED_TE XT	STRING		
id	ID	INTEGER		REQUIRED KEY

Attribute	DB Field	Data Type	SREL References	Flags
last_mod_dt	LAST_MOD_DT	LOCAL_TIME		
LINK_ID	LINK_ID	INTEGER	ES_NODES id	
NODE_ID	NODE_ID	INTEGER		
NODE_SHOR T_DESC	NODE_SHORT_ DESC	STRING		
NODE_TYPE	NODE_TYPE	INTEGER		
PARENT_NOD E_ID	PARENT_NODE _ID	INTEGER		
QUERY_RESP _NUMBER	QUERY_RESP_ NUMBER	INTEGER		
QUERY_RESP _TYPE	QUERY_RESP_T YPE	STRING		
RESLINKID1	RESLINKID1	INTEGER		
RESLINKID2	RESLINKID2	INTEGER		
RESLINKID3	RESLINKID3	INTEGER		
RESLINKID4	RESLINKID4	INTEGER		
RESLINKID5	RESLINKID5	INTEGER		
RESLINKID6	RESLINKID6	INTEGER		
RESLINKID7	RESLINKID7	INTEGER		
RESPONSE1	RESPONSE1	STRING		
RESPONSE2	RESPONSE2	STRING		
RESPONSE3	RESPONSE3	STRING		
RESPONSE4	RESPONSE4	STRING		
RESPONSE5	RESPONSE5	STRING		
RESPONSE6	RESPONSE6	STRING		
RESPONSE7	RESPONSE7	STRING		
ROOT_ID	ROOT_ID	INTEGER	ES_NODES id	
TREE_ID	TREE_ID	INTEGER	SKELETONS id	

ES_RESPONSES Object

The object details are as follows:

Associated Table: ES_RESPONSES

Factories: default

REL_ATTR: id

Common Name: RESPONSE_LINK_TEXT

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		REQUIRED KEY
last_mod_dt	LAST_MOD_DT	LOCAL_TIME		
PARENT_NOD E_ID	PARENT_NODE _ID	INTEGER	ES_NODES id	
RESPONSE_LI NK_ID	RESPONSE_LI NK_ID	INTEGER	ES_NODES id	
RESPONSE_LI NK_ORDER	RESPONSE_LI NK_ORDER	INTEGER		
RESPONSE_LI NK_TEXT	RESPONSE_LI NK_TEXT	STRING		

ES_SESSIONS Object

The object details are as follows:

Associated Table: ES_SESSIONS

Factories: default

REL_ATTR: id

Common Name: EXTERNAL_ID

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
COMMENT_TE XT	COMMENT_TEX T	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
EVALUATION	EVALUATION	INTEGER		
EXTERNAL_ID	EXTERNAL_ID	STRING		
id	ID	INTEGER		REQUIRED KEY
last_mod_dt	LAST_MOD_DT	LOCAL_TIME		
PATH_IDS	PATH_IDS	STRING		
PATH_QAS	PATH_QAS	STRING		
SESSION_ID	SESSION_ID	INTEGER		
TREE_ID	TREE_ID	INTEGER	ES_NODES id	

event_log Object

The object details are as follows:

Associated Table: event_log

Factories: default

REL_ATTR: id

Common Name: sd_obj_type

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
event	event	INTEGER	event_type id	
id	id	INTEGER		UNIQUE REQUIRED KEY
kd	kd_id	INTEGER	SKELETONS id	
log_time	log_time	LOCAL_TIME		
millitime	millitime	INTEGER		
numdata1	numdata1	INTEGER		
numdata2	numdata2	INTEGER		
sd_obj_id	sd_obj_id	INTEGER		
sd_obj_type	sd_obj_type	STRING	e	

Attribute	DB Field	Data Type	SREL References	Flags
session	session	INTEGER	session_log id	
textdata1	textdata1	STRING		
textdata2	textdata2	STRING		

event_type Object

The object details are as follows:

Associated Table: event_type

Factories: default

REL_ATTR: id

Common Name: description

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
code	code	STRING		UNIQUE REQUIRED S_KEY
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		

evt Object Object

The object details are as follows:

Associated Table: Events

Factories: default

REL_ATTR: persistent_id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
condition	condition	STRING	splmac persid	
delete_flag	del	INTEGER	actbool enum	REQUIRED
delay_time	delay_time	DURATION		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_dt	last_mod_dt	LOCAL_TIME		
modulo_time	modulo_time	DURATION		REQUIRED
obj_type	obj_type	STRING		
on_done_flag	on_done_flag	INTEGER		REQUIRED
persid	persid	STRING		
sym	sym	STRING		REQUIRED unique S_KEY
urgency	urgency	INTEGER		
user_settime	user_settime	INTEGER		REQUIRED
user_smag	user_smag	STRING		
violate_on_false	violate_on_false	INTEGER		
violate_on_true	violate_on_true	INTEGER		
work_shift	work_shift	STRING	bpwshft persid	

evtdly Object

The object details are as follows:

Associated Table: Event_Delay

Factories: default

REL_ATTR: persistent_id

Common Name: group_name

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
act_delay	act_delay	DURATION		
cancel_time	cancel_time	LOCAL_TIME		
create_time	create_time	LOCAL_TIME		
delay_type	delay_type	INTEGER	evtdlytp enum	
eff_delay	eff_delay	DURATION		
group_name	group_name	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_dt	last_mod_dt	LOCAL_TIME		
obj_id	obj_id	STRING		REQUIRED S_KEY
persid	persid	STRING		
start_time	start_time	LOCAL_TIME		
start_userid	start_userid	UUID	ca_contact uuid	
status_flag	status_flag	INTEGER		
stop_time	stop_time	LOCAL_TIME		
stop_userid	stop_userid	UUID	ca_contact uuid	
support_lev	support_lev	STRING	srv_desc code	

evtdlytp Object

The object details are as follows:

Associated Table: Event_Delay_Type

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
enum	enum	INTEGER		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
description	nx_desc	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

ext_entity_map Object

The object details are as follows:

Associated Table: External_Entity_Map

Factories: default, sd_chg_map

REL_ATTR: persistent_id

Common Name: xschema_code

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
int1_rsrved	int1_rsrved	INTEGER		
int2_rsrved	int2_rsrved	INTEGER		
int3_rsrved	int3_rsrved	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
int4_rsrved	int4_rsrved	INTEGER		
int5_rsrved	int5_rsrved	INTEGER		
int6_rsrved	int6_rsrved	INTEGER		
lstr1_rsrved	lstr1_rsrved	STRING		
lstr2_rsrved	lstr2_rsrved	STRING		
ob_persid	ob_persid	STRING		
ob_type	ob_type	STRING		
persid	persid	STRING		
str1_rsrved	str1_rsrved	STRING		
str2_rsrved	str2_rsrved	STRING		
xentity_id	xentity_id	STRING		REQUIRED
xschema_code	xschema_code	STRING		REQUIRED
xschema_ver	xschema_ver	INTEGER		REQUIRED

fmgrp Object

The object details are as follows:

Associated Table: Form_Group

Factories: default, all_fmgrp

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	

Attribute	DB Field	Data Type	SREL References	Flags
last_mod_dt	last_mod_dt	LOCAL_TIME		
sym	sym	STRING		REQUIRED S_KEY

g_chg_ext Object

The object details are as follows:

Associated Table: Global_Change_Extension

Factories: default

REL_ATTR: id

Common Name: chg_ref_num

Function Group: change_mgr

Attribute	DB Field	Data Type	SREL References	Flags
active	active_flag	INTEGER	actrbool enum	REQUIRED
affected_contact	affected_contact	UUID	ca_contact uuid	REQUIRED
assignee	assignee	UUID	ca_contact uuid	
category	category	STRING		
chg_ref_num	chg_ref_num	STRING		REQUIRED
close_date	close_date	LOCAL_TIME		
global_queue_id	global_queue_id	INTEGER	g_queue_names id	
group	group_id	UUID		
id	id	INTEGER		UNIQUE REQUIRED KEY
impact	impact	INTEGER	impact enum	
last_mod_dt	last_mod_dt	LOCAL_TIME		REQUIRED
open_date	open_date	LOCAL_TIME		REQUIRED
priority	priority	INTEGER	pri enum	REQUIRED

[g_chg_queue Object](#)

Attribute	DB Field	Data Type	SREL References	Flags
remote_id	remote_id	INTEGER		REQUIRED S_KEY
requestor	requestor	UUID	ca_contact uuid	REQUIRED
status	status	STRING	chgstat code	REQUIRED
summary	summary	STRING		

[g_chg_queue Object](#)

The object details are as follows:

Associated Table: Global_Change_Queue

Factories: default

REL_ATTR: id

Common Name: chg_ref_num

Function Group: change_mgr

Attribute	DB Field	Data Type	SREL References	Flags
active	active_flag	INTEGER	actrbool enum	REQUIRED
affected_contact	affected_contact	UUID		REQUIRED
assignee	assignee	UUID		
category	category	STRING		
chg_ref_num	chg_ref_num	STRING		REQUIRED
close_date	close_date	LOCAL_TIME		
global_queue_id	global_queue_id	INTEGER	g_queue_names id	
group	group_id	UUID		
id	id	INTEGER		UNIQUE REQUIRED KEY
impact	impact	INTEGER	impact enum	

Attribute	DB Field	Data Type	SREL References	Flags
last_mod_dt	last_mod_dt	LOCAL_TIME		REQUIRED
open_date	open_date	LOCAL_TIME		REQUIRED
priority	priority	INTEGER	pri enum	REQUIRED
remote_id	remote_id	INTEGER		REQUIRED S_KEY
remote_sys_id	remote_sys_id	INTEGER	g_srvr remote_sys_id	REQUIRED S_KEY
requestor	requestor	UUID		REQUIRED
status	status	STRING	chgstat code	REQUIRED
summary	summary	STRING		

g_cnt Object

The object details are as follows:

Associated Table: Global_Contact

Factories: default

REL_ATTR: id

Common Name: combo_name

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
contact_num	contact_num	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
email_address	email_address	STRING		
first_name	first_name	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_dt	last_mod_dt	LOCAL_TIME		
last_name	last_name	STRING		

[g_cr_ext Object](#)

Attribute	DB Field	Data Type	SREL References	Flags
location	loc_id	UUID		
middle_name	middle_name	STRING		
organization	org_id	UUID		
pri_phone_number	pri_phone_number	STRING		
remote_id	remote_id	UUID		REQUIRED S_KEY
remote_sys_id	remote_sys_id	INTEGER	g_srvr remote_sys_id	REQUIRED S_KEY
userid	userid	STRING		

[g_cr_ext Object](#)

The object details are as follows:

Associated Table: Global_Request_Extension

Factories: default

REL_ATTR: id

Common Name: ref_num

Function Group: call_mgr

Attribute	DB Field	Data Type	SREL References	Flags
active	active_flag	INTEGER	actrbool enum	REQUIRED
assignee	assignee	UUID	ca_contact uuid	
category	category	STRING		
close_date	close_date	LOCAL_TIME		
customer	customer	UUID	ca_contact uuid	REQUIRED
global_queue_id	global_queue_id	INTEGER	g_queue_names id	
group	group_id	UUID		
id	id	INTEGER		UNIQUE REQUIRED KEY

Attribute	DB Field	Data Type	SREL References	Flags
impact	impact	INTEGER	impact enum	
last_mod_dt	last_mod_dt	LOCAL_TIME		REQUIRED
open_date	open_date	LOCAL_TIME		REQUIRED
priority	priority	INTEGER	pri enum	REQUIRED
ref_num	ref_num	STRING		REQUIRED
remote_id	remote_id	INTEGER		REQUIRED S_KEY
status	status	STRING	cr_stat code	REQUIRED
summary	summary	STRING		
type	type	STRING	crt code	

g_cr_queue Object

The object details are as follows:

Associated Table: Global_Request_Queue

Factories: default

REL_ATTR: id

Common Name: ref_num

Function Group: call_mgr

Attribute	DB Field	Data Type	SREL References	Flags
active	active_flag	INTEGER	actrbool enum	REQUIRED
assignee	assignee	UUID		
category	category	STRING		
close_date	close_date	LOCAL_TIME		
customer	customer	UUID		REQUIRED
global_queue_id	global_queue_id	INTEGER	g_queue_names id	
group	group_id	UUID		
id	id	INTEGER		UNIQUE REQUIRED KEY

[g_iss_ext Object](#)

Attribute	DB Field	Data Type	SREL References	Flags
impact	impact	INTEGER	impact enum	
last_mod_dt	last_mod_dt	LOCAL_TIME		REQUIRED
open_date	open_date	LOCAL_TIME		REQUIRED
priority	priority	INTEGER	pri enum	REQUIRED
ref_num	ref_num	STRING		REQUIRED
remote_id	remote_id	INTEGER		REQUIRED S_KEY
remote_sys_id	remote_sys_id	INTEGER	g_srvr remote_sys_id	REQUIRED S_KEY
status	status	STRING	cr_stat code	REQUIRED
summary	summary	STRING		
type	type	STRING	crt code	

[g_iss_ext Object](#)

The object details are as follows:

Associated Table: Global_Issue_Extension

Factories: default

REL_ATTR: id

Common Name: ref_num

Function Group: issue_mgr

Attribute	DB Field	Data Type	SREL References	Flags
active	active_flag	INTEGER	actrbool enum	REQUIRED
assignee	assignee	UUID	ca_contact uuid	
category	category	STRING		
close_date	close_date	LOCAL_TIME		
global_queue_id	global_queue_id	INTEGER	g_queue_name s id	
group	group_id	UUID		

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
impact	impact	INTEGER	impact enum	
last_mod_dt	last_mod_dt	LOCAL_TIME		REQUIRED
open_date	open_date	LOCAL_TIME		REQUIRED
priority	priority	INTEGER	pri enum	REQUIRED
product	product	INTEGER	product id	
ref_num	ref_num	STRING		REQUIRED
remote_id	remote_id	INTEGER		REQUIRED S_KEY
requestor	requestor	UUID	ca_contact uuid	REQUIRED
status	status	STRING	issstat code	REQUIRED
summary	summary	STRING		

g_iss_queue Object

The object details are as follows:

Associated Table: Global_Issue_Queue

Factories: default

REL_ATTR: id

Common Name: ref_num

Function Group: issue_mgr

Attribute	DB Field	Data Type	SREL References	Flags
active	active_flag	INTEGER	actrbool enum	REQUIRED
assignee	assignee	UUID		
category	category	STRING		
close_date	close_date	LOCAL_TIME		
global_queue_id	global_queue_id	INTEGER	g_queue_names id	
group	group_id	UUID		

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
impact	impact	INTEGER	impact enum	
last_mod_dt	last_mod_dt	LOCAL_TIME		REQUIRED
open_date	open_date	LOCAL_TIME		REQUIRED
priority	priority	INTEGER	pri enum	REQUIRED
product	product	INTEGER		
ref_num	ref_num	STRING		REQUIRED
remote_id	remote_id	INTEGER		REQUIRED S_KEY
remote_sys_id	remote_sys_id	INTEGER	g_srvr remote_sys_id	REQUIRED S_KEY
requestor	requestor	UUID		REQUIRED
status	status	STRING	issstat code	REQUIRED
summary	summary	STRING		

g_loc Object

The object details are as follows:

Associated Table: Global_Location

Factories: default

REL_ATTR: id

Common Name: name

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
del	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_dt	last_mod_dt	LOCAL_TIME		
loc_name	loc_name	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
remote_id	remote_id	UUID		REQUIRED S_KEY
remote_sys_i_d	remote_sys_i_d	INTEGER	g_srvr remote_sys_i_d	REQUIRED S_KEY

g_org Object

The object details are as follows:

Associated Table: Global_Organization

Factories: default

REL_ATTR: id

Common Name: name

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_dt	last_mod_dt	LOCAL_TIME		
name	org_name	STRING		
remote_id	remote_id	UUID		REQUIRED S_KEY
remote_sys_i_d	remote_sys_i_d	INTEGER	g_srvr remote_sys_i_d	REQUIRED S_KEY

g_prod Object

The object details are as follows:

Associated Table: Global_Product

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_dt	last_mod_dt	LOCAL_TIME		
remote_id	remote_id	INTEGER		REQUIRED S_KEY
remote_sys_id	remote_sys_id	INTEGER	g_srvr remote_sys_id	REQUIRED S_KEY
sym	sym	STRING		

g_qname Object

The object details are as follows:

Associated Table: Global_Queue_Names

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
sym	sym	STRING		UNIQUE REQUIRED

g_tblmap Object

The object details are as follows:

Associated Table: Global_Table_Map

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
map_definition	map_definition	STRING		REQUIRED
sym	sym	STRING		UNIQUE REQUIRED

g_tblrule Object

The object details are as follows:

Associated Table: Global_Table_Rule

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
addl_query	addl_query	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
last_sync_dt	last_sync_dt	LOCAL_TIME		
reoccur_inter	reoccur_inter	DURATION		
v	v			
sched	sched	STRING	bpwshft persid	
sym	sym	STRING		UNIQUE REQUIRED S_KEY
table_map	table_map	INTEGER	g_tbl_map id	

g_srvrs Object

The object details are as follows:

Associated Table: Global_Servers

Factories: default

REL_ATTR: remote_sys_id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
chg_prefix	chg_prefix	STRING		
cr_prefix	cr_prefix	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
global_name	global_name	STRING		UNIQUE REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
is_master	is_master	INTEGER	bool_tab enum	
iss_prefix	iss_prefix	STRING		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
remote_sys_id	remote_sys_id	INTEGER		UNIQUE REQUIRED
slump_addr	slump_addr	STRING		
sym	sym	STRING		UNIQUE REQUIRED
web_protocol	web_protocol	STRING		
web_server	web_server	STRING		
web_server_port	web_server_port	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
web_url	web_url	STRING		

gl_code Object

The object details are as follows:

Associated Table: ca_resource_gl_code

Factories: default

REL_ATTR: id

Common Name: name

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
creation_date	creation_date	LOCAL_TIM E		
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIM E		
exclude_registration	exclude_registration	integer		
id	id	INTEGER		UNIQUE REQUIRED KEY
delete_flag	inactive	integer	actbool enum	
last_mod	last_update_da te	LOCAL_TIM E		
last_update_us er	last_update_us er	STRING		
name	name	STRING		
version_numbe r	version_numbe r	integer		

grc Object

The object details are as follows:

Associated Table: ca_resource_class

Factories: default

REL_ATTR: id

Common Name: type

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
creation_date	creation_date	LOCAL_TIME		
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		
exclude_registration	exclude_registration	integer		
family	family_id	INTEGER	ca_resource_family_id	
id	id	INTEGER		UNIQUE REQUIRED KEY
delete_flag	inactive	integer	actbool enum	
last_mod	last_update_date	LOCAL_TIME		
last_update_user	last_update_user	STRING		
type	name	STRING		
parent_id	parent_id	INTEGER		
nsm_class	usp_nsm_class	INTEGER	buscls_id	
version_number	version_number	integer		

grp_loc Object

The object details are as follows:

Associated Table: Group_Loc

Factories: default

REL_ATTR: persistent_id

Common Name: lpid

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		
lpid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

grpmem Object

The object details are as follows:

Associated Table: Group_Member

Factories: default

REL_ATTR: persistent_id

Common Name:

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
group	group_id	UUID		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
manager_flag	manager_flag	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
member	member	UUID	ca_contact uuid	REQUIRED
notify_flag	notify_flag	INTEGER		

hier Object

The object details are as follows:

Associated Table: Asset_Assignment

Factories: default

REL_ATTR: id

Common Name: license_num

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
child	hier_child	byte(16)	ca_owned_reso urce uuid	REQUIRED S_KEY
license_num	hier_license_	nvarchar(40)	num	
log_date	hier_log_dat	INTEGER		REQUIRED
parent	hier_parent	byte(16)	ca_owned_reso urce uuid	REQUIRED S_KEY
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	byte(16)	ca_contact uuid	
last_mod_dt	last_mod_dt	INTEGER		
persid	persid	nvarchar(30)		

imp Object

The object details are as follows:

Associated Table: Impact

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
enum	enum	INTEGER		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
description	nx_desc	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY
value	value	INTEGER		

INDEX_DOC_LINKS Object

The object details are as follows:

Associated Table: INDEX_DOC_LINKS

Factories: default

REL_ATTR: id

Common Name: RELATIONAL_ID

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Document ID	DOC_ID	INTEGER	SKELETONS id	
id	ID	INTEGER		REQUIRED KEY
Category ID	INDEX_ID	INTEGER	O_INDEXES id	
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Relational ID	RELATIONAL_ID	STRING		

intfc Object

The object details are as follows:

Associated Table: Interface

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
code	code	STRING		UNIQUE REQUIRED
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	

Attribute	DB Field	Data Type	SREL References	Flags
last_mod_dt	last_mod_dt	LOCAL_TIME		
desc	nx_desc	STRING		
persid	persid	STRING		
sym	sym	STRING		UNIQUE REQUIRED

iss Object

The object details are as follows:

Associated Table: Issue

Factories: default

REL_ATTR: persistent_id

Common Name: ref_num

Function Group: issue_mgr

Attribute	DB Field	Data Type	SREL Reference	Flags
description	description	STRING	s	
actions	actions	STRING		
active_flag	active_flag	INTEGER		REQUIRED
actual_comp_dat	actual_comp_dat	LOCAL_TIME		
e	e			
actual_cost	actual_cost	INTEGER		
actual_total_time	actual_total_time	DURATION		
affected_contact	affected_contact	UUID	ca_contact uuid	
assignee	assignee	UUID		
backout_plan	backout_plan	STRING		
call_back_date	call_back_date	LOCAL_TIME		
call_back_flag	call_back_flag	INTEGER		
category	category	STRING	isscat code	
cawf_procid	cawf_procid	STRING		

Attribute	DB Field	Data Type	SREL Reference	Flags
			s	
close_date	close_date	LOCAL_TIME		
created_via	created_via	INTEGER	interface id	
effort	effort	STRING		
est_comp_date	est_comp_date	LOCAL_TIME		
est_cost	est_cost	INTEGER		
est_total_time	est_total_time	DURATION		
flag1	flag1	INTEGER		
flag2	flag2	INTEGER		
flag3	flag3	INTEGER		
flag4	flag4	INTEGER		
flag5	flag5	INTEGER		
flag6	flag6	INTEGER		
group	group_id	UUID		
id	id	INTEGER		UNIQUE REQUIRED KEY
impact	impact	INTEGER	impact enum	
justification	justification	STRING		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
log_agent	log_agent	UUID	ca_contact uuid	REQUIRED
macro_predicted _violation	macro_predict_viol	INTEGER		
need_by	need_by	LOCAL_TIME		
open_date	open_date	LOCAL_TIME		
organization	organization	UUID	ca_organization uuid	

Attribute	DB Field	Data Type	SREL Reference	Flags
			s	
parent	parent	STRING	issue persistent_id	
persistent_id	persid	STRING		
person_contacting	person_contacting	INTEGER	person id	
predicted_sla_violation	predicted_sla_violation	INTEGER		
priority	priority	INTEGER	pri enum	REQUIRED
product	product	INTEGER	product id	
ref_num	ref_num	STRING		UNIQUE REQUIRED S_KEY
reporting_method	reporting_method	INTEGER	repmethod id	
requestor	requestor	UUID	ca_contact_uuid	REQUIRED
resolve_date	resolve_date	LOCAL_TIME		
rootcause	rootcause	INTEGER	rootcause id	
service_date	service_date	LOCAL_TIME		
service_num	service_num	STRING		
slaViolation	slaViolation	INTEGER		
start_date	start_date	LOCAL_TIME		
status	status	STRING	issstat code	
string1	string1	STRING		
string2	string2	STRING		
string3	string3	STRING		
string4	string4	STRING		
string5	string5	STRING		
string6	string6	STRING		
summary	summary	STRING		

Attribute	DB Field	Data Type	SREL Reference	Flags
			s	
support_lev	support_lev	STRING	srv_desc code	
template_name	template_name	STRING	iss_templat e template_n ame	
type_of_contact	type_of_contact	INTEGER	toc id	
user1	user1	STRING		
user2	user2	STRING		
user3	user3	STRING		

iss_prp Object

The object details are as follows:

Associated Table: Issue_Property

Factories: default

REL_ATTR: id

Common Name: label

Function Group: issue_mgr

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
label	label	STRING		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
owning_iss	owning_iss	STRING	issue persistent_id	REQUIRED
persid	persid	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
required	required	INTEGER	bool_tab enum	
sample	sample	STRING		
sequence	sequence	INTEGER		REQUIRED
value	value	STRING		

iss_tpl Object

The object details are as follows:

Associated Table: Iss_Template

Factories: default

REL_ATTR: template_name

Common Name: template_name

Function Group: issue_mgr

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TI ME		
persid	persid	STRING		
quick_tmpl_ty pe	quick_tmpl_ty pe	INTEGER	quick_tmpl_type enum	REQUIRED
template	template	STRING	issue persistent_id	
template_clas s	template_clas s	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
template_name	template_name	STRING		UNIQUE REQUIRED S_KEY

iss_wf Object

The object details are as follows:

Associated Table: Issue_Workflow_Task

Factories: default

REL_ATTR: id

Common Name: description

Function Group: issue_mgr

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
actual_duration	actual_duration	DURATION		
asset	asset	UUID	ca_owned_resource uuid	
assignee	assignee	UUID	ca_contact uuid	
completion_date	completion_date	LOCAL_TIME		
cost	cost	INTEGER		
creator	creator	UUID	ca_contact uuid	
date_created	date_created	LOCAL_TIME		
del	del	INTEGER		REQUIRED
done_by	done_by	UUID	ca_contact uuid	
est_completion_date	est_comp_date	LOCAL_TIME		

Attribute	DB Field	Data Type	SREL References	Flags
est_cost	est_cost	INTEGER		
est_duration	est_duration	DURATION		
group	group_id	UUID		
group_task	group_task	INTEGER		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIM E		
object_id	object_id	STRING		REQUIRED
object_type	object_type	STRING		REQUIRED
persid	persid	STRING		
sequence	sequence	INTEGER		REQUIRED
start_date	start_date	LOCAL_TIM E		
status	status	STRING	tskstat code	
support_lev	support_lev	STRING	srv_desc code	
task	task	STRING	tskty code	REQUIRED
wf_template	wf_template	INTEGER	wftpl id	

issalg Object

The object details are as follows:

Associated Table: Issue_Act_Log

Factories: default

REL_ATTR: id

Common Name: description

Function Group: issue_mgr

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
action_desc	action_desc	STRING		
analyst	analyst	UUID	ca_contact uuid	
id	id	INTEGER		UNIQUE REQUIRED KEY
internal	internal	INTEGER		
issue_id	issue_id	STRING	issue persistent_id	
knowledge_se ssion	knowledge_se ssion	STRING		
knowledge_to ol	knowledge_to ol	STRING		
last_mod_dt	last_mod_dt	LOCAL_TIM E		
persid	persid	STRING		
system_time	system_time	LOCAL_TIM E		
time_spent	time_spent	DURATION		
time_stamp	time_stamp	LOCAL_TIM E		
type	type	STRING	act_type code	

isscat Object

The object details are as follows:

Associated Table: Issue_Category

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
assignee	assignee	UUID		
auto_assign	auto_assign	INTEGER		
cawf_defid	cawf_defid	STRING		
children_ok	children_ok	INTEGER		REQUIRED
code	code	STRING		UNIQUE REQUIRED S_KEY
del	del	INTEGER		REQUIRED
group_id	group_id	UUID		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact_uuid	
last_mod_dt	last_mod_dt	LOCALTIME		
organization	organization	UUID	ca_organization_uuid	
owning_contract	owning_contrac	INTEGER	svc_contract_id	
persistent_id	persid	STRING		
schedule	schedule	INTEGER		
service_type	service_type	STRING	srv_desc_code	
survey	survey	INTEGER	survey_tpl_id	

Attribute	DB Field	Data Type	SREL References	Flags
sym	sym	STRING		REQUIRED S_KEY

isscat_grp Object

The object details are as follows:

Associated Table: Isscat_Group

Factories: default

REL_ATTR: persistent_id

Common Name: lpid

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		
lpid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

isscat_loc Object

The object details are as follows:

Associated Table: Isscat_Loc

Factories: default

REL_ATTR: persistent_id

Common Name: Ipid

Function Group:

Attribut e	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		
Ipid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

isscat_workshift Object

The object details are as follows:

Associated Table: Isscat_Workshift

Factories: default

REL_ATTR: persistent_id

Common Name: Ipid

Function Group:

Attribut e	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		
Ipid	l_persid	STRING		

Attribut	DB Field	Data Type	SREL References	Flags
e				
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

issstat Object

The object details are as follows:

Associated Table: Issue_Status

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
active	active	INTEGER		
code	code	STRING		UNIQUE REQUIRED S_KEY
del	del	INTEGER		REQUIRED
hold	hold	INTEGER		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
resolved	resolved	INTEGER		
sym	sym	STRING		REQUIRED

job_func Object

The object details are as follows:

Associated Table: ca_job_function

Factories: default

REL_ATTR: id

Common Name: name

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
creation_date	creation_date	LOCAL_TIME		
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		
description	description	STRING		
exclude_registration	exclude_registration	integer		
id	id	INTEGER		UNIQUE REQUIRED KEY
delete_flag	inactive	integer	actbool enum	
last_mod	last_update_date	LOCAL_TIME		
last_update_user	last_update_user	STRING		
name	name	STRING		
version_number	version_number	integer		

KCAT Object

The object details are as follows:

Associated Table: O_INDEXES

Factories: default

REL_ATTR: id

Common Name: CAPTION

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Description	DESCRIPTION	STRING		
Author ID	AUTHOR_ID	UUID	ca_contact uuid	
Caption	CAPTION	STRING		
Default Document Template	DOC_TEMPLATE	INTEGER	CI_DOC_TEMPLATES id	
Has Children	HAS_CHILDREN	INTEGER		
Has Documents	HAS_DOCS	INTEGER		
id	ID	INTEGER		KEY
Keywords	KEYWORDS	STRING		
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Owner ID	OWNER_ID	UUID	ca_contact uuid	
Parent ID	PARENT_ID	INTEGER	O_INDEXES id	
PERMISSION_IN DEX_ID	PERMISSION_IN DEX_ID	INTEGER	O_INDEXES id	
READ_PGROUP	READ_PGROUP	INTEGER	P_GROUPS id	
Relational ID	RELATIONAL_ID	STRING		
Subject Expert ID	SUBJECT_EXPERT_ID	UUID	ca_contact uuid	
Workflow Template	WF_TEMPLATE	INTEGER	CI_WF_TEMPLATES id	
WRITE_PGROUP	WRITE_PGROUP	INTEGER	P_GROUPS id	

KD Object

The object details are as follows:

Associated Table: SKELETONS

Factories: default

REL_ATTR: id

Common Name: TITLE

Function Group: kd

Attribute	DB Field	Data Type	SREL References	Flags
Document State	ACTIVE_STATE	INTEGER		
Document State Date	ACTIVE_STATE_DATE	LOCAL_TIME		
Assignee ID	ASSIGNEE_ID	UUID	ca_contact uuid	
Author ID	AUTHOR_ID	UUID	ca_contact uuid	
FAQ Rating	BU_RESULT	REAL		
Created Via	CREATED_VIA	INTEGER		
Creation Date	CREATION_DATE	LOCAL_TIME		
Task ID	CURRENT_ACTION_ID	INTEGER	CI_ACTIONS id	
Custom 1	CUSTOM1	STRING		
Custom 2	CUSTOM2	STRING		
Custom 3	CUSTOM3	STRING		
Custom 4	CUSTOM4	STRING		
Custom 5	CUSTOM5	STRING		
Custom Num 1	CUSTOM_NUM1	REAL		
Custom Num 2	CUSTOM_NUM2	REAL		
Document Template ID	DOC_TEMPLATE_ID	INTEGER	CI_DOC_TEMPLATE id	
Document Type ID	DOC_TYPE_ID	INTEGER	CI_DOC_TYPES id	
Document Version	DOC_VERSION	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
Expiration Date	EXPIRATION_DATE	LOCAL_TIME		
	E			
EXPIRE_NOTIFICATION_SENT	EXPIRE_NOTIFICATION_SENT	INTEGER		
Decision Tree Root ID	EXT_DOC_ID	INTEGER		
FULLWORDS	FULLWORDS	STRING		
Hits	HITS	INTEGER		
id	ID	INTEGER		KEY
INDEXED	INDEXED	INTEGER		
Inherit Permissions Flag	INHERIT_PERMISSIONS_FLAG	INTEGER		
Initiator	INITIATOR	STRING		
Initiator ID	INITIATOR_ID	UUID	ca_contact uuid	
Inherit Permissions from Category ID	KD_PERMISSION_INDEX_ID	INTEGER	O_INDEXES id	
Last Accepted Date	LAST_ACCEPTED_DATE	LOCAL_TIME		
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Locked By ID	LOCKED_BY_ID	UUID	ca_contact uuid	
Modify Date	MODIFY_DATE	LOCAL_TIME		
Notes	NOTES	STRING		
Owner ID	OWNER_ID	UUID	ca_contact uuid	
Parent Request	PARENT_CR	STRING	call_req persid	
Parent Issue	PARENT_ISS	STRING	issue persistent_id	
Primary Category	PRIMARY_INDEX	INTEGER	O_INDEXES id	
Workflow Priority ID	PRIORITY_ID	INTEGER	CI_PRIORITIES id	
Problem	PROBLEM	STRING		
Published Date	PUBLISHED_DATE	LOCAL_TIME		

Attribute	DB Field	Data Type	SREL References	Flags
READ_PGROUP	READ_PGROUP	INTEGER	P_GROUPS id	
Resolution Text	RESOLUTION	STRING		
Resolution Length	RESOLUTION_LEN	INTEGER		
Short Resolution	RESOLUTION_SH	STRING		
Review Date	REVIEW_DATE	LOCAL_TIME		
Solution Count	SD_ACCEPTED_HI_TS	INTEGER		
Asset ID	SD_ASSET_ID	UUID	ca_owned_resource uuid	
Impact ID	SD_IMPACT_ID	INTEGER	impact enum	
Priority ID	SD_PRIORITY_ID	INTEGER	pri enum	
Product ID	SD_PRODUCT_ID	INTEGER	product id	
Root Cause ID	SD_ROOTCAUSE_ID	INTEGER	rootcause id	
Severity ID	SD_SEVERITY_ID	INTEGER	sevrty enum	
Urgency ID	SD_URGENCY_ID	INTEGER	urgncy enum	
SHORTWORDS	SHORTWORDS	STRING		
Start Date	START_DATE	LOCAL_TIME		
Status ID	STATUS_ID	INTEGER	CI_STATUSES id	
Subject Expert ID	SUBJECT_EXPERT_ID	UUID	ca_contact uuid	
Summary	SUMMARY	STRING		
Title	TITLE	STRING		
User Defined ID	USER_DEF_ID	STRING		
Version Comment	VER_COMMENT	STRING		
Version Count	VER_COUNT	INTEGER		
Version Cross Reference ID	VER_CROSS_REF_ID	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
Workflow Template	WF_TEMPLATE	INTEGER	CI_WF_TEMPLATES id	
WORD_COUNT _TOTAL	WORD_COUNT_T OTAL	INTEGER		
WORDCOUNT	WORDCOUNT	INTEGER		
WORDCOUNTS	WORDCOUNTS	STRING		
WORDORDERS	WORDORDERS	STRING		
WORDPLACES	WORDPLACES	STRING		
WORDSPANS	WORDSPANS	STRING		
WRITE_PGROU P	WRITE_PGROUP	INTEGER	P_GROUPS id	

KD_ATTMNT Object

The object details are as follows:

Associated Table: KD_ATTMNT

Factories: default

REL_ATTR: id

Common Name:

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
ATTMNT_ID	ATTMNT_ID	INTEGER	attmnt id	
DOC_ID	DOC_ID	INTEGER	SKELETONS id	
id	ID	INTEGER		KEY
last_mod_dt	LAST_MOD_DT	LOCAL_TIME		

kdlinks Object

The object details are as follows:

Associated Table: kdlinks

Factories: default

REL_ATTR: persistent_id

Common Name: sd_obj_type

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Request Linked	cr	STRING	call_req persid	
id	ID	INTEGER		KEY
Issue Linked	iss	STRING	issue persistent_id	
Document	kd	INTEGER	SKELETONS id	
Analyst	last_mod_by	UUID	ca_contact uuid	
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Link Type	link_type	INTEGER		
Ticket ID	sd_obj_id	INTEGER		
Ticket Type	sd_obj_type	STRING		

kmlrel Object

The object details are as follows:

Associated Table: Knowledge_Lrel_Table

Factories: default

REL_ATTR: persistent_id

Common Name: lpid

Function Group:

Attribut e	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY

Attribut e	DB Field	Data Type	SREL References	Flags
lattr	l_attr	STRING		
lpid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

KT_REPORT_CARD Object

The object details are as follows:

Associated Table: KT_REPORT_CARD

Factories: default

REL_ATTR: id

Common Name: SUBJECT_ID

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
AVERAGE_EFFECTI VENESS_RATING	AVERAGE_EFFECTIVE NESS_RATING	INTEGER		
creation_date	creation_date	INTEGER		
creation_user	creation_user	STRING		
DOCUMENTS_PUBL ISHED	DOCUMENTS_PUBLIS HED	INTEGER		
DOCUMENTS_SUB MITTED	DOCUMENTS_SUBMI TTED	INTEGER		
id	ID	INTEGER		KEY
last_update_date	last_update_date	INTEGER		
last_update_user	last_update_user	STRING		
ORG_STATISTICS	ORG_STATISTICS	INTEGER		
PAST_DAYS	PAST_DAYS	INTEGER		
SUBJECT_ID	SUBJECT_ID	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
TOTAL_HITS	TOTAL_HITS	INTEGER		
TOTAL_SOLUTION_ COUNT	TOTAL_SOLUTION_C OUNT	INTEGER		

kwd Object

The object details are as follows:

Associated Table: Knowledge_Keywords

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_d t	last_mod_d t	LOCAL_TIME		
persid	persid	STRING		
sym	sym	STRING		REQUIRED S_KEY

loc Object

The object details are as follows:

Associated Table: ca_location

Factories: default

REL_ATTR: id

Common Name: name

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
address1	address_1	STRING		
address2	address_2	STRING		
address3	address_3	STRING		
address4	address_4	STRING		
address5	address_5	STRING		
address6	address_6	STRING		
city	city	STRING		
description	comment	STRING		
contact_address	contact_address_flag	integer		
	_flag			
country	country	integer	ca_country_id	
county	county	STRING		
creation_date	creation_date	LOCAL_TIME		
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		
exclude_registration	exclude_registration	integer		
fax_number	fax_number	STRING		
geo_coord_type	geo_coord_type	integer		
geo_coords	geo_coords	STRING		
delete_flag	inactive	integer	actbool enum	

Attribute	DB Field	Data Type	SREL References	Flags
last_mod	last_update_date	LOCAL_TIME		
last_update_use r	last_update_user	STRING		
name	location_name	STRING		
id	location_uuid	UUID		UNIQUE REQUIRED REDKEY
mail_address_1	mail_address_1	STRING		
mail_address_2	mail_address_2	STRING		
mail_address_3	mail_address_3	STRING		
mail_address_4	mail_address_4	STRING		
mail_address_5	mail_address_5	STRING		
mail_address_6	mail_address_6	STRING		
pri_phone_number er	pri_phone_number	STRING		
primary_contact _uuid	primary_contact_uuid	UUID		
site	site_id	integer	ca_site id	
state	state	integer	ca_state_pro vince id	
version_number	version_number	integer		
zip	zip	STRING		

LONG_TEXTS Object

The object details are as follows:

Associated Table: LONG_TEXTS

Factories: default

REL_ATTR: persistent_id

Common Name: REF_PERSID

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Chunk	ACTUAL_TEXT	STRING		
Cunk Order	CNT_ORDER	INTEGER		
id	ID	INTEGER		KEY
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Document ID	REF_PERSID	STRING		

lr Object

The object details are as follows:

Associated Table: Notify_Log_Header

Factories: default

REL_ATTR: persistent_id

Common Name: msg_hdr

Function Group: notify

Attribute	DB Field	Data Type	SREL References	Flags
cmth_used	cmth_used	INTEGER	ct_mth id	
cntxt_obj	cntxt_obj	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY

Attribute	DB Field	Data Type	SREL References	Flags
last_mod	last_mod	LOCAL_TIME		
nlh_ack_by	nlh_ack_by	LOCAL_TIME		
ack_time	nlh_ack_time	DURATION		
contact	nlh_c_address ee	UUID	ca_contact uuid	
nlh_c_alias	nlh_c_alias	UUID		
notify_metho d	nlh_cm_metho d	INTEGER	noturg enum	
email_addresses	nlh_email	STRING		
end_date	nlh_end	LOCAL_TIME		
msg_hdr	nlh_hdr	STRING		
msg_text	nlh_msg	STRING		
msg_html	nlh_msg_html	STRING		
pri_event	nlh_pri	INTEGER		
start_date	nlh_start	LOCAL_TIME		REQUIRED
status	nlh_status	INTEGER		
activity_notify	nlh_transition	INTEGER		
notify_type	nlh_type	INTEGER		
msg_ack	nlh_user_ack	STRING		

lrel1 Object

The object details are as follows:

Associated Table: Lrel_Table

Factories: default

REL_ATTR: persistent_id

Common Name: lpid

Function Group:

Attribut e	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		
lpid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

macro Object

The object details are as follows:

Associated Table: Spell_Macro

Factories: default

REL_ATTR: persistent_id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
usr_string1	fragment	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_dt	last_mod_dt	LOCAL_TIME		
lock_object	lock_object	INTEGER		REQUIRED
msg_html	msg_html	STRING		
ob_type	ob_type	STRING		REQUIRED
persid	persid	STRING		
sym	sym	STRING		REQUIRED UNIQUE
type	type	STRING	splmactp persid	REQUIRED
usr_integer1	usr_integer1	INTEGER		
usr_integer2	usr_integer2	INTEGER		
usr_integer3	usr_integer3	INTEGER		
usr_string2	usr_string2	STRING		
usr_string3	usr_string3	STRING		
usr_string4	usr_string4	STRING		

macro_type Object

The object details are as follows:

Associated Table: Spell_Macro_Type

Factories: default, edit_macros

REL_ATTR: code, persistent_id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
arg_list	arg_list	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
code	code	STRING		UNIQUE REQUIRED
delete_flag	del	INTEGER	actbool enum	REQUIRED
display_name	display_name	STRING		
execute_script	execute_script	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_dt	last_mod_dt	LOCAL_TI ME		
lock_object_flag	lock_object_flag	INTEGER		
persid	persid	STRING		
sym	sym	STRING		REQUIRED
tech_desc	tech_desc	STRING		
validate_script	validate_script	STRING		

mfrmod Object

The object details are as follows:

Associated Table: ca_model_def

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
abbreviation	abbreviation	STRING		
resource_class	class_id	INTEGER	ca_resource_classes	
creation_date	creation_date	LOCAL_TIME		

Attribute	DB Field	Data Type	SREL	Flags
			References	
creation_user	creation_user	STRING		
current_as_of _date	current_as_of _date	integer		
delete_time	delete_time	LOCAL_TIM E		
exclude_regs tration	exclude_regs tration	integer		
family_id	family_id	INTEGER		
delete_flag	inactive	integer	actbool enum	
last_mod	last_update_ date	LOCAL_TIM E		
last_update_ user	last_update_ user	STRING		
manufacturer	manufacturer _uuid	UUID	ca_company company_uuid	
id	model_uuid	UUID		UNIQUE REQUIRED KEY
sym	name	STRING		
operating_sys tem	operating_sys tem	integer		
preferred_sell er_uuid	preferred_sell er_uuid	UUID		
version_num ber	version_num ber	integer		
model_name	description	STRING		

mgs Object

The object details are as follows:

Associated Table: Managed_Survey

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
open_date	create_date	LOCAL_TIME		
active	del	INTEGER	bool_tab enum	REQUIRED
close_date	end_date	LOCAL_TIME		
group	group_id	UUID		
id	id	INTEGER		UNIQUE REQUIRED KEY
initial_method	initial_method	INTEGER	ct_mth id	
initial_msgbod	initial_msgbod	STRING		
y	y			
initial_msgttitle	initial_msgttitle	STRING		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
assignee	owner	UUID	ca_contact uuid	
persid	persid	STRING		
reminder_meth od	reminder_meth od	INTEGER	ct_mth id	
reminder_msg body	reminder_msg body	STRING		
reminder_msigt itle	reminder_msigt itle	STRING		
start_dt	start_date	LOCAL_TIME		

Attribute	DB Field	Data Type	SREL References	Flags
status	status	STRING	mgsstat code	
sym	sym	STRING		UNIQUE REQUIRED S_KEY
tplid	tplid	INTEGER	survey_tpl id	

mgsalg Object

The object details are as follows:

Associated Table: Mgs_Act_Log

Factories: default

REL_ATTR: id

Common Name: description

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
action_desc	action_desc	STRING		
analyst	analyst	UUID	ca_contact uuid	
id	id	INTEGER		UNIQUE REQUIRED KEY
internal	internal	INTEGER		
last_mod_dt	last_mod_dt	LOCAL_TIME		
mgs_id	mgs_id	INTEGER	managed_surv ey id	
persid	persid	STRING		
system_time	system_time	LOCAL_TIME		
time_spent	time_spent	DURATION		
time_stamp	time_stamp	LOCAL_TIME		
type	type	STRING	act_type code	

mgsstat Object

The object details are as follows:

Associated Table: Mgs_Status

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
active	active	INTEGER		
code	code	STRING		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
hold	hold	INTEGER		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
modified_date	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
sym	sym	STRING		REQUIRED

NOTIFICATION Object

The object details are as follows:

Associated Table: NOTIFICATION

Factories: default

REL_ATTR: id

Common Name: ALT_EMAIL

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Email Address	ALT_EMAIL	STRING		
Contact ID	ANALYST_ID	UUID	ca_contact uuid	
Document ID	DOC_ID	INTEGER		
id	ID	INTEGER		REQUIRED KEY
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
KT Notification Level	NTF_LEVEL	INTEGER		

notque Object

The object details are as follows:

Associated Table: Queued_Notify

Factories: default

REL_ATTR: persistent_id

Common Name: msg_title

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
cmth_override	cmth_override	INTEGER		
context_persid	context_persi_d	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
internal	internal	INTEGER		
msg_ack	msg_ack	STRING		
msg_body	msg_body	STRING		
msg_body_html	msg_body_ht	STRING	ml	
msg_title	msg_title	STRING		
notify_level	notify_level	INTEGER		
persid	persid	STRING		
transition_pt	transition_pt	INTEGER		

noturg Object

The object details are as follows:

Associated Table: Notification_Urgency

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
enum	enum	INTEGER		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
description	nx_desc	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

nr Object

The object details are as follows:

Associated Table: ca_owned_resource, usp_owned_resource

Factories: default

REL_ATTR: id

Common Name: name

Function Group: inventory

ca_owned_resource Table

Attribute	DB Field	Data Type	SREL References	Flags
acquire_date	acquire_date	LOCAL_TIME		
asset_source_uuid	asset_source_uuid	UUID		
loc_cabinet	cabinet_location	STRING		
company_bought_for_uuid	company_bought_for_uuid	UUID	ca_company	
expense_code	cost_center	integer	ca_resource_cost_center_id	
creation_date	creation_date	LOCAL_TIME		
creation_system	creation_system	STRING		
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		
department	department	INTEGER	ca_resource_department_id	
exclude_registration	exclude_registration	integer		
loc_floor	floor_location	STRING		
gl_code	gl_code	integer		

Attribute	DB Field	Data Type	SREL References	Flags
system_name	host_name	STRING		
delete_flag	inactive	integer	actbool enum	
install_date	installation_date	LOCAL_TIME		
alarm_id	ip_address	STRING		
last_mod	last_update_date	LOCAL_TIME		
last_mod_by	last_update_user	STRING		
license_number	license_infor	STRING		
license_uuid	license_uuid	UUID		
location	location_uuid	UUID	ca_location	
mac_address	mac_address	STRING		
repair_org	maintenance_org_uuid	UUID	ca_organization	
vendor_repair	maintenance_vendor_uuid	UUID	ca_company	
manufacturer	manufacture	UUID	ca_company	
model	model_uuid	UUID	ca_model_def	
operating_system	operating_system	integer		
org_bought_for_uuid	org_bought_for_uuid	UUID	ca_organization	
id	own_resource_uuid	UUID		UNIQUE REQUIRED KEY
product_version	product_version	STRING		
purchase_order_id	purchase_order_id	STRING		
requisition_id	requisition_id	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
resource_aliases	resource_aliases	STRING		
class	resource_class	integer	ca_resource_class id	
resource_contact_act	resource_contact_uuid	UUID	ca_contact uuid	
description	resource_description	STRING		
family	resource_family	integer	ca_resource_family id	
name	resource_name	STRING		
resource_owner_uuid	resource_owner_uuid	UUID	ca_contact uuid	
asset_count	resource_quantity	integer		
status	resource_status	integer	ca_resource_status id	
asset_num	resource_tag	STRING		
service_org	responsible_organization_uuid	UUID	ca_organization uuid	
vendor_restore	responsible_vendor_uuid	UUID	ca_company company_uuid	
loc_room	room_location	STRING		
serial_number	serial_number	STRING		
loc_shelf	shelf_location	STRING		
loc_slot	slot_location	STRING		
status_date	status_date	LOCAL_TIME		
supplier	supply_vendor_uuid	UUID	ca_company company_uuid	
version_number	version_number	integer		

usp_owned_resource Table

Attribute	DB Field	Data Type	SREL References	Flags
argis_id	nr_argis_id	STRING		
bm_rep	nr_bm_rep	INTEGER	busrep id	
bm_label	nr_bmlabel	STRING		
bm_status	nr_bms	INTEGER	busstat status_no	
expiration_date	nr_exp_dt	LOCAL_TIME		
financial_num	nr_financial_id	STRING		
contact_1	nr_nx_ref_1	UUID	ca_contact uuid	
contact_2	nr_nx_ref_2	UUID	ca_contact uuid	
contact_3	nr_nx_ref_3	UUID	ca_contact uuid	
smag_1	nr_nx_string1	STRING		
smag_2	nr_nx_string2	STRING		
smag_3	nr_nx_string3	STRING		
smag_4	nr_nx_string4	STRING		
smag_5	nr_nx_string5	STRING		
smag_6	nr_nx_string6	STRING		
priority	nr_pr_id	INTEGER	pri enum	
name_type	nr_prim_skt_id	INTEGER		
service_type	nr_service_type	STRING	srv_desc code	
sla	nr_sla_id	INTEGER		
warranty_end	nr_wrty_end_dt	LOCAL_TIME		
warranty_start	nr_wrty_st_dt	LOCAL_TIME		
id	owned_resource_uuid	UUID		

nr_com Object

The object details are as follows:

Associated Table: NR_Comment

Factories: default

REL_ATTR: id

Common Name: writer_name

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
attr_name	attr_name	STRING		
log	com_comment	STRING		
log_date	com_dt	LOCAL_TIME		REQUIRED
asset_id	com_par_id	UUID	ca_owned_res ource uuid	REQUIRED S_KEY
writer_name	com_userid	STRING		REQUIRED S_KEY
id	id	INTEGER		UNIQUE REQUIRED KEY
new_value	new_value	STRING		
old_value	old_value	STRING		
writer_id	writer_id	UUID		

nrf Object

The object details are as follows:

Associated Table: ca_resource_family

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
creation_date	creation_date	LOCAL_TIME		
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		
exclude_registration	exclude_registration	integer		
id	id	INTEGER		UNIQUE REQUIRED KEY
delete_flag	inactive	integer	actbool	
			enum	
include_reconciliation	include_reconciliation	integer		
last_update_date	last_update_date	LOCAL_TIME		
last_update_user	last_update_user	STRING		
sym	name	STRING		
extension_name	table_extension_name	STRING		
version_number	version_number	integer		

ntfl Object

The object details are as follows:

Associated Table: Notify_Object_Attr

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
modified_date	last_mod_dt	LOCALTIME		
object_attr	object_attr	STRING		
object_type	object_type	STRING		
persid	persid	STRING		
sym	sym	STRING		REQUIRED

O_COMMENTS Object

The object details are as follows:

Associated Table: O_COMMENTS

Factories: default

REL_ATTR: id

Common Name: COMMENT_TEXT

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Comment Text	COMMENT_TEXT	STRING		
Comment Timestamp	COMMENT_TIMESTAMP	LOCAL_TIME		
Document ID	DOC_ID	INTEGER	SKELETONS id	
Email Address	EMAIL_ADDRESS	STRING		
id	ID	INTEGER		KEY
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Contact ID	USER_ID	UUID	ca_contact uuid	
Contact Name	USER_NAME	STRING		
Version Count	VER_COUNT	INTEGER		

O_EVENTS Object

The object details are as follows:

Associated Table: O_EVENTS

Factories: default

REL_ATTR: id

Common Name: EVENT_NAME

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Change details	ACTION	STRING		
Document ID	ENTITY_ID	INTEGER	SKELETONS id	
Change Type	EVENT_NAME	STRING		
Date and Time	EVENT_TIMESTAMP	LOCAL_TIME		
P				
id	ID	INTEGER		KEY
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
For future use	VER_COUNT	INTEGER		
Change Type Enum	WF_ACTION_ID	INTEGER		
User ID	WF_USER_ID	UUID	ca_contact uuid	

opsys Object

The object details are as follows:

Associated Table: ca_resource_operating_system

Factories: default

REL_ATTR: id

Common Name: name

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
creation_date	creation_date	LOCAL_TIME		
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		
exclude_registration	exclude_registration	integer		
id	id	INTEGER		UNIQUE REQUIRED KEY
delete_flag	inactive	integer	actbool enum	
last_mod	last_update_date	LOCAL_TIME		
last_update_user	last_update_user	STRING		
name	name	STRING		
version_number	version_number	integer		

options Object

The object details are as follows:

Associated Table: Options

Factories: default

REL_ATTR: persistent_id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
action	action	INTEGER		
action_status	action_status	STRING		
app_name	app_name	STRING		
default_value	default_value	STRING		
deinstall_script	deinstall_script	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
error_msg	error_msg	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
install_script	install_script	STRING		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
option_name	option_name	STRING		
persid	persid	STRING		
readme	readme	STRING		
sequence	sequence	INTEGER		
sym	sym	STRING		REQUIRED
validation	validation	STRING		
value	value	STRING		
value_active	value_active	INTEGER		

org Object

The object details are as follows:

Associated Table: Internal_Organization

Factories: default

REL_ATTR: id

Common Name: name

Function Group: inventory

ca_organization Table

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
org_num	abbreviation	STRING		
alt_phone_cc	alt_phone_cc	integer		
alt_phone	alt_phone_nu mber	STRING		
comment	comment	STRING		
company	company_uuid	UUID	ca_company company_uui d	
contact	contact_uuid	UUID	ca_contact uuid	
billing_code	cost_center	integer	ca_resource_ cost_center id	
creation_dat e	creation_date	LOCAL_TIME		
creation_use r	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		
email_addr	email_address	STRING		
exclude_regi stration	exclude_registration	integer		
fax_cc	fax_cc	integer		
fax_phone	fax_number	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	inactive	integer	actbool enum	
last_mod	last_update_date	LOCAL_TIME		
last_update_user	last_update_us	STRING		
location	location_uuid	UUID	ca_location	
			location_uuid	
name	org_name	STRING		
id	organization_uid	UUID		UNIQUE REQUIRED KEY
pemail_addr	pager_email_address	STRING		
parent_org_uuid	parent_org_uuid	UUID		
pri_phone_cc	pri_phone_cc	integer		
phone_number	pri_phone_number	STRING		
version_number	version_number	integer		

usp_organization Table

Attribute	DB Field	Data Type	SREL References	Flags
iorg_assigned_sv	iorg_assigned_sv	INTEGER		
svr	r			
service_type	iorg_service_type	STRING	srv_desc	code
last_mod	last_mod	LOCAL_TIME		
id	organization_uuid	UUID		
owning_contract	owning_contract	INTEGER	svc_contract	id

P_GROUPS Object

The object details are as follows:

Associated Table: P_GROUPS

Factories: default

REL_ATTR: id

Common Name: GRP_LIST_K

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
GRP_LIST	GRP_LIST	STRING		
GRP_LIST_KEY	GRP_LIST_KEY	STRING		
id	ID	INTEGER		KEY
last_mod_dt	LAST_MOD_DT	LOCAL_TIME		

pcat Object

The object details are as follows:

Associated Table: Prob_Category

Factories: default

REL_ATTR: persistent_id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
assignee	assignee	UUID		
auto_assign	auto_assign	INTEGER		
cr_flag	cr_flag	INTEGER		
delete_flag	del	INTEGER	actbool enum	REQUIRED
group	group_id	UUID		

[pcat_grp Object](#)

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
in_flag	in_flag	INTEGER		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
organization	organization	UUID	ca_organization uuid	
owning_cont ract	owning_cont ract	INTEGER	svc_contract id	
persistent_id	persid	STRING		
pr_flag	pr_flag	INTEGER		
schedule	schedule	INTEGER		
service_type	service_type	STRING	srv_desc code	
survey	survey	INTEGER	survey_tpl id	
sym	sym	STRING		REQUIRED S_KEY
tcode	tcode	INTEGER		

[pcat_grp Object](#)

The object details are as follows:

Associated Table: Pcat_Group

Factories: default

REL_ATTR: persistent_id

Common Name: lpid

Function Group:

Attribut e	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		

Attribut e	DB Field	Data Type	SREL References	Flags
lpid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

pcat_loc Object

The object details are as follows:

Associated Table: Pcat_Loc

Factories: default

REL_ATTR: persistent_id

Common Name: lpid

Function Group:

Attribut e	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		
lpid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

pcat_workshift Object

The object details are as follows:

Associated Table: Pcat_Workshift

Factories: default

REL_ATTR: persistent_id

Common Name: lpid

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		
lpid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

perscnt Object

The object details are as follows:

Associated Table: Person_Contacting

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY

Attribute	DB Field	Data Type	SREL References	Flags
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

position Object

The object details are as follows:

Associated Table: ca_job_title

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
creation_date	creation_dat e	LOCAL_TIME		
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		
exclude_regi stration	exclude_regi stration	integer		
id	id	INTEGER		UNIQUE REQUIRED KEY
delete_flag	inactive	integer	actbool enum	
last_mod_dt	last_update_ date	LOCAL_TIME		
last_mod_by	last_update_ user	STRING		
sym	name	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
version_num ber	version_num ber	integer		

pri Object

The object details are as follows:

Associated Table: Priority

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
enum	enum	INTEGER		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TI ME		
description	nx_desc	STRING		
service_type	service_type	STRING	srv_desc code	
sym	sym	STRING		UNIQUE REQUIRED S_KEY

prod Object

The object details are as follows:

Associated Table: Product

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TI ME		
persid	persid	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

prp Object

The object details are as follows:

Associated Table: Property

Factories: default

REL_ATTR: id

Common Name: value

Function Group: change_mngr

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
label	label	STRING		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
object_id	object_id	INTEGER	chg id	REQUIRED
object_type	object_type	STRING		REQUIRED
persid	persid	STRING		
property	property	INTEGER	prptpl id	
required	required	INTEGER	bool_tab enum	
sample	sample	STRING		
sequence	sequence	INTEGER		REQUIRED
value	value	STRING		

prptpl Object

The object details are as follows:

Associated Table: Property_Template

Factories: default

REL_ATTR: id

Common Name: label

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
code	code	STRING		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
label	label	STRING		REQUIRED
last_mod_by	last_mod_by	UUID	ca_contact uuid	
modified_date	last_mod_dt	LOCAL_TIME		
object_attrname	object_attrname	STRING		REQUIRED
object_attrval	object_attrval	INTEGER		
object_type	object_type	STRING		REQUIRED
persid	persid	STRING		
required	required	INTEGER		REQUIRED
sample	sample	STRING		
sequence	sequence	INTEGER		REQUIRED

quick_tpl_types Object

The object details are as follows:

Associated Table: Quick_Template_Types

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
enum	enum	INTEGER		UNIQUE REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
description	nx_desc	STRING		
persid	persid	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
sym	sym	STRING		

rc Object

The object details are as follows:

Associated Table: Rootcause

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

response Object

The object details are as follows:

Associated Table: Response

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
chg_flag	chg_flag	INTEGER		S_KEY
cr_flag	cr_flag	INTEGER		S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
in_flag	in_flag	INTEGER		S_KEY
iss_flag	iss_flag	INTEGER		S_KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
pr_flag	pr_flag	INTEGER		S_KEY
response	response	STRING		
response_owner	response_owner	UUID	ca_contact uuid	S_KEY
sym	sym	STRING		REQUIRED S_KEY

rev_bool Object

The object details are as follows:

Associated Table: Reverse_Boolean_Table

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER		REQUIRED
enum	enum	INTEGER		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
desc	nx_desc	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

rptm Object

The object details are as follows:

Associated Table: Rpt_Meth

Factories: default

REL_ATTR: persistent_id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
default_pag	def_pg_len	STRING		
e_length				
default_out	default_out	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
is_default	is_default	INTEGER		
last_mod_b_y	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
script	script	STRING		
sym	sym	STRING		REQUIRED

rptmeth Object

The object details are as follows:

Associated Table: Reporting_Method

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

rrf Object

The object details are as follows:

Associated Table: Remote_Ref

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
arch_type	arch_type	STRING		
code	code	STRING		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
exec_str	exec_str	STRING		
function_group	function_group	STRING		
up	up			
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
pceexec_str	pceexec_str	STRING		
sym	sym	STRING		REQUIRED

rss Object

The object details are as follows:

Associated Table: ca_resource_status

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
creation_date	creation_date	LOCAL_TIME		
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		
exclude_registration	exclude_registration	integer		
id	id	INTEGER		UNIQUE REQUIRED KEY
delete_flag	inactive	integer	actbool enum	
last_update_date	last_update_date	LOCAL_TIME		
last_update_user	last_update_user	STRING		
sym	name	STRING		
version_number	version_number	integer		

sapolicy Object

The object details are as follows:

Associated Table: SA_Policy

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL Reference	Flags
description	description	STRING		
attachment	access_atmnt	INTEGER		
data_query	access_data	INTEGER		
knowledge_opp	access_knowl edge	INTEGER		
object_insertion	access_object _ins	INTEGER		
object_update	access_object _upd	INTEGER		
ticket_insertion	access_ticket _ins	INTEGER		
allow_imperonate	allow_imperonate	INTEGER	actbool enum	REQUIRED
code	code	STRING		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
ext_appl	ext_appl	INTEGER		
id	id	INTEGER		UNIQUE REQUIRED KEY
is_default	is_default	INTEGER		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIM E		

Attribute	DB Field	Data Type	SREL Reference	Flags
			s	
persid	persid	STRING		
proxy_contact	proxy_contact	UUID	ca_contact uuid	
public_key	pub_key	STRING		
state	state	INTEGER		
sym	sym	STRING		REQUIRED

saprobtyp Object

The object details are as follows:

Associated Table: SA_Prob_Type

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
code	code	STRING		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
action	dup_action	INTEGER		
search_interval	dup_interval	DURATION		
id	id	INTEGER		UNIQUE REQUIRED KEY
is_default	is_default	INTEGER		
is_internal	is_internal	INTEGER	actbool enum	
last_mod_by	last_mod_by	UUID	ca_contact uuid	

Attribute	DB Field	Data Type	SREL References	Flags
last_mod_dt	last_mod_dt	LOCAL_TI ME		
owning_policy	owning_policy	INTEGER	sapolicy id	
persid	persid	STRING		
program_outp ut	ret_app_1	STRING		
user_output	ret_usr_1	STRING		
sym	sym	STRING		REQUIRED
template_facto ry	ticket_tmpl_f ac	STRING		
template_id	ticket_tmpl_id	INTEGER		
template_sym	ticket_tmpl_n ame	STRING		

sdsc Object

The object details are as follows:

Associated Table: Service_Desc

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
code	code	STRING		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY

Attribute	DB Field	Data Type	SREL References	Flags
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
owning_contract	owning_cont ract	INTEGER	svc_contract id	
persid	persid	STRING		
rank	rank	INTEGER		
schedule	schedule	STRING	bpwshft persid	
sym	sym	STRING		REQUIRED S_KEY
violation_cost	violation_cos t	INTEGER		

sdsc_map Object

The object details are as follows:

Associated Table: SLA_Contract_Map

Factories: default

REL_ATTR: id

Common Name: map_persid

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
map_contract	map_contrac t	INTEGER	svc_contract id	REQUIRED
map_persid	map_persid	STRING		
map_sdsc	map_sdsc	STRING	srv_desc code	

Attribute	DB Field	Data Type	SREL References	Flags
persid	persid	STRING		

seq Object

The object details are as follows:

Associated Table: Sequence_Control

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
code	code	STRING		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
prefix	prefix	STRING		
suffix	suffix	STRING		
sym	sym	STRING		REQUIRED

session_log Object

The object details are as follows:

Associated Table: session_log

Factories: default

REL_ATTR: id

Common Name:

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
contact	contact	UUID	ca_contact uuid	
id	id	INTEGER		UNIQUE REQUIRED KEY
login_time	login_time	LOCAL_TIME		
logout_time	logout_time	LOCAL_TIME		
policy	policy	INTEGER	sapolicy id	
session_id	session_id	INTEGER		
session_type	session_type	INTEGER	session_type id	
status	status	INTEGER		

session_type Object

The object details are as follows:

Associated Table: session_type

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
sym	sym	STRING		REQUIRED S_KEY

sev Object

The object details are as follows:

Associated Table: Severity

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
enum	enum	INTEGER		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
description	nx_desc	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

SHOW_OBJ Object

The object details are as follows:

Associated Table: SHOW_OBJ

Factories: default

REL_ATTR: id

Common Name: OBJ_PERSID

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Expiration Date	EXPIRE_DATE	LOCAL_TIME		
id	ID	INTEGER		KEY
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Object ID	OBJ_PERSID	STRING		
Password	PWD	STRING		

site Object

The object details are as follows:

Associated Table: ca_site

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: inventory

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
alias	alias	STRING		
contact	contact_uuid	UUID	ca_contact uuid	
creation_date	creation_date	LOCAL_TIME		
creation_user	creation_user	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
delete_time	delete_time	LOCAL_TIME		
exclude_registration	exclude_registration	integer		
id	id	INTEGER		UNIQUE REQUIRED KEY
delete_flag	inactive	integer	actbool enum	
last_mod	last_update_date	LOCAL_TIME		
last_update_use_r	last_update_user	STRING		
name	name	STRING		
version_number	version_number	integer		

slatpl Object

The object details are as follows:

Associated Table: SLA_Template

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
elapsed	elapsed	DURATION		REQUIRED
event	event	STRING	evt persid	
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		

Attribute	DB Field	Data Type	SREL References	Flags
object_type	object_type	STRING		
persid	persid	STRING		
service_type	service_type	STRING	srv_desc code	REQUIRED
sym	sym	STRING		REQUIRED S_KEY

srvr_aliases Object

The object details are as follows:

Associated Table: Server_Aliases

Factories: default

REL_ATTR: id

Common Name: alias_name

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
alias_name	alias_name	STRING		REQUIRED
delete_flag	del	INTEGER	actbool enum	REQUIRED
host_addr	host_addr	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
zone_id	zone_id	INTEGER	srvr_zones id	REQUIRED

srvr_zones Object

The object details are as follows:

Associated Table: Server_Zones

Factories: default

REL_ATTR: id

Common Name: zone_name

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
is_default	is_default	INTEGER	bool_tab enum	
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
zone_name	zone_name	STRING		REQUIRED

state Object

The object details are as follows:

Associated Table: ca_state_province

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
creation_date	creation_date	LOCAL_TIME		
creation_user	creation_user	STRING		
delete_time	delete_time	LOCAL_TIME		

Attribute	DB Field	Data Type	SREL References	Flags
exclude_registration	exclude_registration	integer		
id	id	INTEGER		UNIQUE REQUIRED KEY
delete_flag	inactive	integer	actbool enum	
last_mod_dt	last_update_date	LOCAL_TIME		
last_mod_by	last_update_user	STRING		
sym	symbol	STRING		
version_number	version_number	integer		

survey Object

The object details are as follows:

Associated Table: Survey

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
comment_label	comment_label	STRING		
conclude_text	conclude_text	STRING		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
include_comment	include_comment	INTEGER		REQUIRED

Attribute	DB Field	Data Type	SREL References	Flags
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
comment	nx_comment	STRING		
object_id	object_id	INTEGER		
object_type	object_type	STRING		
persid	persid	STRING		
sym	sym	STRING		S_KEY

svc_contract Object

The object details are as follows:

Associated Table: Service_Contract

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
active	active	INTEGER	actbool enum	
contract_nu m	contract_nu m	STRING		
delete_flag	del	INTEGER	actbool enum	
dflt_chgcat_ st	dflt_chgcat_ st	STRING	srv_desc code	
dflt_cnt_st	dflt_cnt_st	STRING	srv_desc code	
dflt_isscat_st	dflt_isscat_st	STRING	srv_desc code	
dflt_nr_st	dflt_nr_st	STRING	srv_desc code	
dflt_pcat_st	dflt_pcat_st	STRING	srv_desc code	
dflt_pri_st	dflt_pri_st	STRING	srv_desc code	
expiration	expiration	LOCAL_TIME		

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
description	nx_desc	STRING		
service_type	org_svc_type	STRING	srv_desc code	
persid	persid	STRING		
svc_advocate	svc_advocate	UUID	ca_contact uuid	
svc_owner	svc_owner	UUID	ca_contact uuid	
sym	sym	STRING		

svy_atpl Object

The object details are as follows:

Associated Table: Survey_Answer_Template

Factories: default

REL_ATTR: id

Common Name: text

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
owning_survey_question	own_srvy_q	INTEGER	survey_qtpl id	
persid	persid	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
sequence	sequence	INTEGER		REQUIRED
text	txt	STRING		

svy_ans Object

The object details are as follows:

Associated Table: Survey_Answer

Factories: default

REL_ATTR: id

Common Name: text

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
owning_surve	own_srvy_q	INTEGER	survey_question	
ey_question	uestion		id	
persid	persid	STRING		
selected	selected	INTEGER		
sequence	sequence	INTEGER		REQUIRED
text	txt	STRING		

svy_qtpl Object

The object details are as follows:

Associated Table: Survey_Question_Template

Factories: default

REL_ATTR: id

Common Name: text

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
include_qcomment	include_qcomment	INTEGER		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
mult_resp_flag	mult_resp_flag	INTEGER		
owning_survey	owning_survey	INTEGER	survey_tpl id	
persid	persid	STRING		
qcomment_label	qcomment_label	STRING		
resp_required	resp_required	INTEGER		
sequence	sequence	INTEGER		REQUIRED
text	txt	STRING		

svy ques Object

The object details are as follows:

Associated Table: Survey_Question

Factories: default

REL_ATTR: id

Common Name: text

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
include_qcomment	include_qcomment	INTEGER		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
mult_resp_flag	mult_resp_flag	INTEGER		
owning_survey	owning_survey	INTEGER	survey id	
persid	persid	STRING		
qcomment	qcomment	STRING		
qcomment_label	qcomment_label	STRING		
resp_required	resp_required	INTEGER		
response	response	INTEGER		REQUIRED
sequence	sequence	INTEGER		REQUIRED
text	txt	STRING		

svy_tpl Object

The object details are as follows:

Associated Table: Survey_Template

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
comment_label	comment_label	STRING		
Survey Completion Message	conclude_text	STRING		
cycle_counter	cycle_counter	INTEGER		
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
include_comment	include_comment	INTEGER		REQUIRED
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
Submit Cycle	submit_cycle	INTEGER		
Survey Name	sym	STRING		UNIQUE REQUIRED S_KEY
tracking_flag	tracking_flag	INTEGER		
Survey Introduction	description	STRING		

svystat Object

The object details are as follows:

Associated Table: Survey_Stats

Factories: default

REL_ATTR: id

Common Name:

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
cyc_counter	cyc_counter	INTEGER		
cycle	cycle	INTEGER		
delete_flag	del	INTEGER	actbool enum	REQUIRED
eval_counter	eval_counte r	INTEGER		
id	id	INTEGER		UNIQUE REQUIRED KEY
persid	persid	STRING		
sub_counter	sub_counter	INTEGER		
tplid	tplid	INTEGER	survey_tpl id	

svytrk Object

The object details are as follows:

Associated Table: Survey_Tracking

Factories: default

REL_ATTR: id

Common Name: object_type

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
cntid	cntid	UUID	ca_contact uuid	
delete_flag	del	INTEGER	actbool enum	REQUIRED

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
notif_dt	notif_dt	LOCAL_TIME		
object_id	object_id	INTEGER		
object_typ e	object_typ e	STRING		
persid	persid	STRING		
recv_dt	recv_dt	LOCAL_TIME		
status	status	INTEGER		
tplid	tplid	INTEGER	survey_tpl id	

tskstat Object

The object details are as follows:

Associated Table: Task_Status

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
allow_accumul ate	allow_accumulat e	INTEGER		REQUIRED
allow_task_upd ate	allow_task_upd ate	INTEGER		REQUIRED
code	code	STRING		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
do_next_task	do_next_task	INTEGER		REQUIRED
hold	hold	INTEGER		REQUIRED

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
is_internal	is_internal	INTEGER		REQUIRED
last_mod_by	last_mod_by	UUID	ca_contact uuid	
modified_date	last_mod_dt	LOCAL_TI ME		
no_update_ms	no_update_msg	STRING		
persid	persid	STRING		
sym	sym	STRING		REQUIRED
task_complete	task_complete	INTEGER		REQUIRED

tskty Object

The object details are as follows:

Associated Table: Task_Type

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
code	code	STRING		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	

Attribute	DB Field	Data Type	SREL References	Flags
modified_date	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
sym	sym	STRING		REQUIRED S_KEY

tspan Object

The object details are as follows:

Associated Table: Timespan

Factories: default

REL_ATTR: sym

Common Name:

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
code	code	STRING		UNIQUE REQUIRED
end_day	end_day	STRING		
end_hour	end_hour	STRING		
end_minute	end_minute	STRING		
end_month	end_month	STRING		
end_year	end_year	STRING		
id	id	INTEGER		UNIQUE KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
desc	nx_desc	STRING		
start_day	start_day	STRING		
start_hour	start_hour	STRING		
start_minute	start_minute	STRING		
start_month	start_month	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
start_year	start_year	STRING		
sym	sym	STRING		UNIQUE REQUIRED
trigger_day	trigger_day	STRING		
trigger_hour	trigger_hour	STRING		
trigger_minute	trigger_minute	STRING		
trigger_month	trigger_month	STRING		
trigger_year	trigger_year	STRING		

[typecnt Object](#)

The object details are as follows:

Associated Table: Type_Of_Contact

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TI ME		
persid	persid	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY

tz Object

The object details are as follows:

Associated Table: Timezone

Factories: default

REL_ATTR: code

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
code	code	STRING		UNIQUE REQUIRED S_KEY
delete_flag	del	INTEGER	actbool enum	REQUIRED
dst_delta	dst_delta	INTEGER		
end_abs_date	end_abs_date	LOCAL_TIME		
end_day	end_day	INTEGER		
end_mon	end_mon	INTEGER		
end_pos	end_pos	INTEGER		
gmt_delta	gmt_delta	INTEGER		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
start_abs_dat e	start_abs_dat e	LOCAL_TIME		
start_day	start_day	INTEGER		
start_mon	start_mon	INTEGER		
start_pos	start_pos	INTEGER		
sym	sym	STRING		REQUIRED S_KEY

urg Object

The object details are as follows:

Associated Table: Urgency

Factories: default

REL_ATTR: enum

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
enum	enum	INTEGER		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
description	nx_desc	STRING		
sym	sym	STRING		UNIQUE REQUIRED S_KEY
value	value	INTEGER		

USP_PREFERENCES Object

The object details are as follows:

Associated Table: USP_PREFERENCES

Factories: default

REL_ATTR: id

Common Name: ONE_B_SEARCH_ORDER

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
Contact ID	ANALYST_ID	UUID	ca_contact uuid	
Knowledge Categories - Documents Per Page	ARC_DOCS_TO_DISPLAY	INTEGER		
Assignee	ASSIGNEE	INTEGER		
Author	AUTHOR	INTEGER		
FAQ Rating	BU_RESULT	INTEGER		
Mouseover Menus	CLASSIC_RESUL TSET_CONTEXT	INTEGER		
Created Via	CREATED_VIA	INTEGER		
Creation Date	CREATION_DATE	INTEGER		
Current Task	CURRENT_ACTIO N	INTEGER		
Custom 1	CUSTOM1	INTEGER		
Custom 2	CUSTOM2	INTEGER		
Custom 3	CUSTOM3	INTEGER		
Custom 4	CUSTOM4	INTEGER		
Custom 5	CUSTOM5	INTEGER		
Custom Num 1	CUSTOM_NUM1	INTEGER		
Custom Num 2	CUSTOM_NUM2	INTEGER		
Document ID	DOC_ID	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
Document Template	DOC_TEMPLATE	INTEGER		
Document Type	DOC_TYPE	INTEGER		
Document Version	DOC_VERSION	INTEGER		
Expiration Date	EXPIRATION_DATE	INTEGER		
GLOBALSD_ACTIVE_ZONE	GLOBALSD_ACTIVE_ZONE	INTEGER		
Hits	HITS	INTEGER		
id	ID	INTEGER		REQUIRED KEY
INBOX_COUNTR	INBOX_COUNTR	INTEGER		
Initiator	INITIATOR	INTEGER		
Item	ITEM	INTEGER		
KT_REPORT_CARD_APAST_DAYS	KT_REPORT_CARD_PAST_DAYS	INTEGER		
KT_REPORT_CARD_SCREEN_DEFULT	KT_REPORT_CARD_SCREEN_DEFALT	INTEGER		
Last Accepted Date	LAST_ACCEPTED_DATE	INTEGER		
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Modify Date	MODIFY_DATE	INTEGER		
Knowledge Documents View Mode	ONE_B_DOC_VIEW_MODE	INTEGER		
Knowledge Documents Per Page	ONE_B_DOCS_TODISPLAY	INTEGER		
Knowledge Documents Show Details Flag	ONE_B_HIDE_DETAILSTAILS	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
EBR Match Type	ONE_B_MATCH_TYPE	INTEGER		
EBR Search Fields	ONE_B_SEARCH_FIELDS	INTEGER		
EBR Search Order	ONE_B_SEARCH_ORDER	STRING		
EBR Search Type	ONE_B_SEARCH_TYPE	INTEGER		
EBR Word Parts	ONE_B_WORD_PARTS	INTEGER		
Owner	OWNER	INTEGER		
Primary Category	PRIMARY_INDEX	INTEGER		
Workflow Priority	PRIORITY	INTEGER		
Product	PRODUCT	INTEGER		
Published Date	PUBLISHED_DATE	INTEGER		
Review Date	REVIEW_DATE	INTEGER		
Solution Count	SD_ACCEPTED_HITS	INTEGER		
Impact	SD_IMPACT	INTEGER		
Priority	SD_PRIORITY	INTEGER		
Root Cause	SD_ROOTCAUSE	INTEGER		
SD_SEARCH_FILEDS_CR	SD_SEARCH_FILEDS_CR	INTEGER		
SD_SEARCH_ISS	SD_SEARCH_ISS	INTEGER		
Severity	SD_SEVERITY	INTEGER		
Urgency	SD_URGENCY	INTEGER		
Start Date	START_DATE	INTEGER		
Status	STATUS	INTEGER		
Subject Expert	SUBJECT_EXPERT	INTEGER		
User Defined ID	USER_DEF_ID	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
WEB_LAST_LOGI N	WEB_LAST_LOGI N	LOCAL_TIM E		
WEB_POPUP1_H EIGHT	WEB_POPUP1_H EIGHT	INTEGER		
WEB_POPUP1_W IDTH	WEB_POPUP1_W IDTH	INTEGER		
WEB_POPUP2_H EIGHT	WEB_POPUP2_H EIGHT	INTEGER		
WEB_POPUP2_W IDTH	WEB_POPUP2_W IDTH	INTEGER		
WEB_POPUP3_H EIGHT	WEB_POPUP3_H EIGHT	INTEGER		
WEB_POPUP3_W IDTH	WEB_POPUP3_W IDTH	INTEGER		
WEB_POPUP4_H EIGHT	WEB_POPUP4_H EIGHT	INTEGER		
WEB_POPUP4_W IDTH	WEB_POPUP4_W IDTH	INTEGER		
WEB_PREFEREN CES	WEB_PREFEREN CES	INTEGER		
WEB_SUPPRESS _TOUR	WEB_SUPPRESS _TOUR	INTEGER		
WEB_TOOLBAR_ TAB	WEB_TOOLBAR_ TAB	INTEGER		
WF_TEMPLATE	WF_TEMPLATE	INTEGER		

USP_PROPERTIES Object

The object details are as follows:

Associated Table: USP_PROPERTIES

Factories: default

REL_ATTR: id

Common Name: PROPERTY_NAME

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	ID	INTEGER		REQUIRED KEY
Last Modified Date	LAST_MOD_DT	LOCAL_TIME		
Property Default	PROPERTY_DEFAL_LT	STRING		
Property Description	PROPERTY_DESCR	STRING PTION		
Property Name	PROPERTY_NAME	STRING		S_KEY
Property Type	PROPERTY_TYPE	STRING		
Property Value	PROPERTY_VALUE	STRING		

usq Object

The object details are as follows:

Associated Table: User_Query

Factories: default

REL_ATTR: id

Common Name: label

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
expanded	expanded	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
factory	factory	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
label	label	STRING		REQUIRED
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
obj_persid	obj_persid	STRING		
parent	parent	INTEGER	usq id	
persid	persid	STRING		
query	query	STRING	crsq code	
query_set	query_set	INTEGER		
query_type	query_type	INTEGER		
sequence	sequence	INTEGER		REQUIRED

vpt Object

The object details are as follows:

Associated Table: ca_company_type

Factories: default

REL_ATTR: id

Common Name: sym

Function Group: reference

Attribute	DB Field	Data Type	SREL References	Flags
creation_date	creation_date	LOCAL_TIME		
creation_user	creation_user	STRING		
description	description	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	inactive	integer	actbool enum	
last_mod	last_update_date	LOCAL_TIME		
last_update_user	last_update_user	STRING		
sym	name	STRING		
version_number	version_number	integer		

wf Object

The object details are as follows:

Associated Table: Workflow_Task

Factories: default

REL_ATTR: id

Common Name: description

Function Group: change_mgr

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
actual_duration	actual_duration	DURATION		
asset	asset	UUID	ca_owned_resource uuid	
assignee	assignee	UUID	ca_contact uuid	
completion_date	completion_date	LOCAL_TIME		
cost	cost	INTEGER		
creator	creator	UUID	ca_contact uuid	
date_created	date_created	LOCAL_TIME		
delete_flag	del	INTEGER	actbool enum	REQUIRED
done_by	done_by	UUID	ca_contact uuid	
est_completion_date	est_comp_date	LOCAL_TIME		

Attribute	DB Field	Data Type	SREL References	Flags
est_cost	est_cost	INTEGER		
est_duration	est_duration	DURATION		
group	group_id	UUID		
group_task	group_task	INTEGER		REQUIRED
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
chg	object_id	INTEGER	chg id	REQUIRED
object_type	object_type	STRING		REQUIRED
persid	persid	STRING		
sequence	sequence	INTEGER		REQUIRED
start_date	start_date	LOCAL_TIME		
status	status	STRING	tskstat code	
support_lev	support_lev	STRING	srv_desc code	
task	task	STRING	tskty code	REQUIRED
wf_template	wf_template	INTEGER	wftpl id	

wftpl Object

The object details are as follows:

Associated Table: Workflow_Task_Template

Factories: default

REL_ATTR: id

Common Name: id

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
assignee	assignee	UUID	ca_contact uuid	
auto_assign	auto_assign	INTEGER		
delete_flag	del	INTEGER	actbool enum	REQUIRED
deleteable	deleteable	INTEGER		REQUIRED
est_cost	est_cost	INTEGER		
est_duration	est_duration	DURATION		
group	group_id	UUID		
id	id	INTEGER		UNIQUE REQUIRED KEY
last_mod_by	last_mod_by	UUID	ca_contact uuid	
modified_date	last_mod_dt	LOCAL_TIME		
object_attrname	object_attrname	STRING		REQUIRED
object_attrval	object_attrval	INTEGER		REQUIRED
object_type	object_type	STRING		REQUIRED
persid	persid	STRING		
sequence	sequence	INTEGER		REQUIRED
service_type	service_type	STRING	srv_desc code	
task	task	STRING	tskty code	REQUIRED

wftpl_grp Object

The object details are as follows:

Associated Table: Wftpl_Group

Factories: default

REL_ATTR: persistent_id

Common Name: lpid

Function Group:

Attribute	DB Field	Data Type	SREL References	Flags
id	id	INTEGER		UNIQUE REQUIRED KEY
lattr	l_attr	STRING		
lpid	l_persid	STRING		
lseq	l_sql	INTEGER		
rattr	r_attr	STRING		
rpid	r_persid	STRING		
rseq	r_sql	INTEGER		

wrkshft Object

The object details are as follows:

Associated Table: Bop_Workshift

Factories: default

REL_ATTR: persistent_id

Common Name: sym

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
delete_flag	del	INTEGER	actbool enum	REQUIRED
description	description	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY

Attribute	DB Field	Data Type	SREL References	Flags
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
persid	persid	STRING		
sched	sched	STRING		
sym	sym	STRING		REQUIRED

wspcol Object

The object details are as follows:

Associated Table: wspcol

Factories: default

REL_ATTR: id

Common Name: column_name

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		
addl_info	addl_info	STRING		
column_name	column_name	STRING		REQUIRED
dbms_name	dbms_name	STRING		
display_name	display_name	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
is_cluster	is_cluster	INTEGER		
is_descending	is_descending	INTEGER		
is_indexed	is_indexed	INTEGER		
is_local	is_local	INTEGER		
is_REQUIRED	is_REQUIRED	INTEGER		
is_order_by	is_order_by	INTEGER		

Attribute	DB Field	Data Type	SREL References	Flags
is_required	is_required	INTEGER		
is_skey	is_skey	INTEGER		
is_unique	is_unique	INTEGER		
is_write_new	is_write_new	INTEGER		
is_wsp	is_wsp	INTEGER		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
on_ci_set	on_ci_set	STRING		
on_new_default	on_new_default	STRING		
persid	persid	STRING		
schema_name	schema_name	STRING		
status	status	INTEGER		REQUIRED
string_len	string_len	INTEGER		
table_name	table_name	STRING		REQUIRED
type	type	INTEGER		
xrel_table	xrel_table	STRING		

wsptbl Object

The object details are as follows:

Associated Table: wsptbl

Factories: default

REL_ATTR: id

Common Name: table_name

Function Group: admin

Attribute	DB Field	Data Type	SREL References	Flags
description	description	STRING		

Attribute	DB Field	Data Type	SREL References	Flags
common_name	common_name	STRING		
dbms_name	dbms_name	STRING		
display_group	display_group	STRING		
display_name	display_name	STRING		
function_group	function_group	STRING		
id	id	INTEGER		UNIQUE REQUIRED KEY
is_local	is_local	INTEGER		
is_wsp	is_wsp	INTEGER		
last_mod_by	last_mod_by	UUID	ca_contact uuid	
last_mod_dt	last_mod_dt	LOCAL_TIME		
methods	methods	STRING		
persid	persid	STRING		
rel_attr	rel_attr	STRING		
schema_name	schema_name	STRING		
sort_by	sort_by	STRING		
status	status	INTEGER		REQUIRED
table_name	table_name	STRING		REQUIRED
triggers	triggers	STRING		

Appendix C: Table and Object Cross-References

This appendix provides several tables that allow you to easily cross-reference table names, SQL names, and object names. The appendix "Data Element Dictionary" (see page 215) lists a complete definition of the tables in the schema. The appendix "Objects and Attributes" (see page 461) lists the object definitions.

Table to SQL Name and Object

This table provides a cross-reference of each table in the database schema to its corresponding SQL and object names:

Table	SQL Name (AKA)	Object
Access_Levels	acc_lvls	acc_lvls
Access_Type	acctyp	acctyp
Act_Log	act_log	alg
Act_Type	act_type	aty
Act_Type_Assoc	atyp_asc	act_type_assoc
Active_Boolean_Table	actbool	actbool
Active_Reverse_Boolean_Table	actrbool	actrbool
admin_tree	admin_tree	ADMIN_TREE
Am_Asset_Map	am_map	am_asset_map
Animator	anima	ANI
Archive_Purge_History	arcpur_hist	arcpur_hist
Archive_Purge_Rule	arcpur_rule	arcpur_rule
Asset_Assignment	hier	hier
Atomic_Condition	atomic_cond	atomic_cond
Attached_Events	att_evt	atev
Attached_SLA	attached_sla	attached_sla
Attachment	attmnt	attmnt

Table to SQL Name and Object

Table	SQL Name (AKA)	Object
Attachment_Lrel	attmnt_lrel	attmnt_lrel
attmnt_folder	attmnt_folder	attmnt_folder
Attribute_Name	atn	
Audit_Log	audit_log	audlog
Behavior_Template	bhvtpl	bhvtpl
Boolean_Table	bool_tab	bool
Bop_Workshift	bpwshft	wrkshft
BU_TRANS	BU_TRANS	BU_TRANS
Business_Management	busmgt	bmhier
Business_Management_Class	buscls	bmcls
Business_Management_Repository	busrep	bmrep
Business_Management_Repository_Lrel	buslrel	bmlrel
Business_Management_Status	busstat	bms
ca_asset_type	ca_asset_type	
ca_company	ca_company	ca_cmpny
ca_company_type	ca_company_type	vpt
ca_contact	ca_contact	cnt
ca_contact_type	ca_contact_type	ctp
ca_country	ca_country	country
ca_job_function	ca_job_function	job_func
ca_job_title	ca_job_title	position
ca_location	ca_location	loc
ca_model_def	ca_model_def	mfrmod
ca_organization	ca_organization	org
ca_owned_resource	ca_owned_resource	nr
ca_resource_class	ca_resource_class	grc
ca_resource_cost_center	ca_resource_cost_center	cost_cntr
	r	

Table	SQL Name (AKA)	Object
ca_resource_department	ca_resource_departmen t	dept
ca_resource_family	ca_resource_family	nrf
ca_resource_gl_code	ca_resource_gl_code	gl_code
ca_resource_operating_s ystem	ca_resource_operating_ system	opsys
ca_resource_status	ca_resource_status	rss
ca_schema_info	ca_schema_info	
ca_site	ca_site	site
ca_state_province	ca_state_province	state
Call_Req	call_req	cr
Call_Req_Type	crt	crt
Call_Solution	crsol	crsol
Change_Act_Log	chgalg	chgalg
Change_Category	chgcat	chgcat
Change_Request	chg	chg
Change_Status	chgstat	chgstat
Chg_Template	chg_template	chg_tpl
Chgcat_Group	ccat_grp	chgcat_grp
Chgcat_Loc	ccat_loc	chgcat_loc
Chgcat_Workshift	ccat_wrkshft	chgcat_workshift
CI_ACTIONS	CI_ACTIONS	CI_ACTIONS
CI_ACTIONS_ALTERNAT E	CI_ACTIONS_ALTERNAT E	CI_ACTIONS_ALTERNATE
CI_BOOKMARKS	CI_BOOKMARKS	CI_BOOKMARKS
CI_DOC_LINKS	CI_DOC_LINKS	CI_DOC_LINKS
CI_DOC_TEMPLATES	CI_DOC_TEMPLATES	CI_DOC_TEMPLATES
CI_DOC_TYPES	CI_DOC_TYPES	CI_DOC_TYPES
CI_PRIORITIES	CI_PRIORITIES	CI_PRIORITIES
CI_STATUSES	CI_STATUSES	CI_STATUSES
CI_WF_TEMPLATES	CI_WF_TEMPLATES	CI_WF_TEMPLATES

Table	SQL Name (AKA)	Object
Column_Name	cn	
Contact_Method	ct_mth	cmth
Controlled_Table	ctab	ctab
Cr_Call_Timers	crctmr	ctimer
Cr_Status	cr_stat	crs
Cr_Stored_Queries	crsq	crsq
Cr_Template	cr_template	cr_tpl
D_PAINTER	D_PAINTER	
Delegation_Server	dlgtsrv	dlgsrvr
Document_Repository	doc_rep	doc_rep
Domain	dmn	dmn
Domain_Constraint	dcon	dcon
Domain_Constraint_Type	dcon_typ	dcon_typ
EBR_ACRONYMS	EBR_ACRONYMS	EBR_ACRONYMS
EBR_FULLTEXT	EBR_FULLTEXT	EBR_FULLTEXT
EBR_FULLTEXT_ADMIN	EBR_FULLTEXT_ADMIN	EBR_FULLTEXT_ADMIN
EBR_FULLTEXT_SD	EBR_FULLTEXT_SD	EBR_FULLTEXT_SD
EBR_FULLTEXT_SD_ADMIN	EBR_FULLTEXT_SD_ADMIN	EBR_FULLTEXT_SD_ADMIN
EBR_INDEXING_QUEUE	EBR_INDEXING_QUEUE	EBR_INDEXING_QUEUE
EBR_KEYWORDS	EBR_KEYWORDS	EBR_KEYWORDS
EBR_LOG	EBR_LOG	EBR_LOG
EBR_METRICS	EBR_METRICS	EBR_METRICS
EBR_NOISE_WORDS	EBR_NOISE_WORDS	EBR_NOISE_WORDS
EBR_PATTERNS	EBR_PATTERNS	EBR_PATTERNS
EBR_PREFIXES	EBR_PREFIXES	EBR_PREFIXES
EBR_PROPERTIES	EBR_PROPERTIES	EBR_PROPERTIES
EBR_SUBSTITUTIONS	EBR_SUBSTITUTIONS	EBR_SUBSTITUTIONS
EBR_SUFFIXES	EBR_SUFFIXES	EBR_SUFFIXES
EBR_SYNONYMS	EBR_SYNONYMS	EBR_SYNONYMS

Table	SQL Name (AKA)	Object
EBR_SYNONYMS_ADM	EBR_SYNONYMS_ADM	EBR_SYNONYMS_ADM
ES_CONSTANTS	ES_CONSTANTS	ES_CONSTANTS
ES_NODES	ES_NODES	ES_NODES
ES_RESPONSES	ES_RESPONSES	ES_RESPONSES
ES_SESSIONS	ES_SESSIONS	ES_SESSIONS
Event_Delay	evt_dly	evtdly
Event_Delay_Type	evtdlytp	evtdlytp
event_log	event_log	event_log
event_type	event_type	event_type
Events	evt	evt
ext_appl	ext_appl	
External_Entity_Map	xent_map	ext_entity_map
Form_Group	frmgrp	fmgrp
Global_Change_Extension	g_chg_ext	g_chg_ext
Global_Change_Queue	g_chg_queue	g_chg_queue
Global_Contact	g_contact	g_cnt
Global_Issue_Extension	g_iss_ext	g_iss_ext
Global_Issue_Queue	g_iss_queue	g_iss_queue
Global_Location	g_loc	g_loc
Global_Organization	g_org	g_org
Global_Product	g_product	g_prod
Global_Queue_Names	g_queue_names	g_qname
Global_Request_Extension	g_req_ext	g_cr_ext
Global_Request_Queue	g_req_queue	g_cr_queue
Global_Servers	g_srvr	g_srvrs
Global_Table_Map	g_tbl_map	g_tblmap
Global_Table_Rule	g_tbl_rule	g_tblrule
Group_Loc	grp_loc	grp_loc
Group_Member	grpmem	grpmem

Table to SQL Name and Object

Table	SQL Name (AKA)	Object
Impact	impact	imp
INDEX_DOC_LINKS	INDEX_DOC_LINKS	INDEX_DOC_LINKS
Interface	interface	intfc
Iss_Template	iss_template	iss_tpl
Isscat_Group	icat_grp	isscat_grp
Isscat_Loc	icat_loc	isscat_loc
Isscat_Workshift	icat_wrkshft	isscat_workshift
Issue	issue	iss
Issue_Act_Log	issalg	issalg
Issue_Category	isscat	isscat
Issue_Property	issprp	iss_prp
Issue_Status	issstat	issstat
Issue_Workflow_Task	isswf	iss_wf
KD_ATTMNT	KD_ATTMNT	KD_ATTMNT
kdlinks	kdlinks	kdlinks
Key_Control	kc	
Knowledge_Keywords	km_kword	kwd
Knowledge_Lrel_Table	km_lrel	kmlrel
KT_REPORT_CARD	KT_REPORT_CARD	KT_REPORT_CARD
LONG_TEXTS	LONG_TEXTS	LONG_TEXTS
Lrel_Table	lrel	lrel1
Managed_Survey	managed_survey	mgs
Mgs_Act_Log	mgsalg	mgsalg
Mgs_Status	mgsstat	mgsstat
Note_Board	cnote	cnote
NOTIFICATION	NOTIFICATION	NOTIFICATION
Notification_Urgency	noturg	noturg
Notify_Log_Header	not_log	lr
Notify_Object_Attr	ntfl	ntfl
NR_Comment	nr_com	nr_com

Table	SQL Name (AKA)	Object
O_COMMENTS	O_COMMENTS	O_COMMENTS
O_EVENTS	O_EVENTS	O_EVENTS
O_INDEXES	O_INDEXES	KCAT
Options	options	options
P_GROUPS	P_GROUPS	P_GROUPS
Pcat_Group	pcat_grp	pcat_grp
Pcat_Loc	pcat_loc	pcat_loc
Pcat_Workshift	pcat_wrkshft	pcat_workshift
Person_Contacting	perscon	perscnt
Priority	pri	pri
Prob_Category	prob_ctg	pcat
Product	product	prod
Property	prp	prp
Property_Template	prptpl	prptpl
Queued_Notify	not_que	notque
Quick_Template_Types	quick_tpl_types	quick_tpl_types
Remote_Ref	rem_ref	rrf
Reporting_Method	repmeth	rptmeth
Req_Property	cr_prp	cr_prp
Req_Property_Template	cr_prptpl	cr_prptpl
Response	response	response
Reverse_Boolean_Table	rbootab	rev_bool
Rootcause	rootcause	rc
Rpt_Meth	rptmth	rptm
SA_Policy	sapolicy	sapolicy
SA_Prob_Type	saprobtyp	saprobtyp
Sequence_Control	seqctl	seq
Server_Aliases	srvr_aliases	srvr_aliases
Server_Zones	srvr_zones	srvr_zones
Service_Contract	svc_contract	svc_contract

Table to SQL Name and Object

Table	SQL Name (AKA)	Object
Service_Desc	srv_desc	sdsc
session_log	session_log	session_log
session_type	session_type	session_type
Severity	sevrty	sev
SHOW_OBJ	SHOW_OBJ	SHOW_OBJ
SKELETONS	SKELETONS	KD
SLA_Contract_Map	sdsc_map	sdsc_map
SLA_Template	slatpl	slatpl
Spell_Macro	splmac	macro
Spell_Macro_Type	splmactp	macro_type
SQL_Script	sql_tab	
Survey	survey	survey
Survey_Answer	survey_answer	svy_ans
Survey_Answer_Templat e	survey_atpl	svy_atpl
Survey_Question	survey_question	svy_ques
Survey_Question_Templat ate	survey_qtpl	svy_qtpl
Survey_Stats	survey_statistics	svystat
Survey_Template	survey_tpl	svy_tpl
Survey_Tracking	survey_tracking	svytrk
Table_Name	tn	
Task_Status	tskstat	tskstat
Task_Type	tskty	tskty
Timespan	tspan	tspan
Timezone	tz	tz
Transition_Points	nottrn	
Type_Of_Contact	toc	typecnt
Urgency	urgncy	urg
User_Query	usq	usq
usp_contact	usp_contact	cnt

Table	SQL Name (AKA)	Object
usp_organization	usp_organization	org
usp_owned_resource	usp_owned_resource	nr
USP_PREFERENCES	USP_PREFERENCES	USP_PREFERENCES
USP_PROPERTIES	USP_PROPERTIES	USP_PROPERTIES
Wftpl_Group	wftpl_grp	wftpl_grp
Workflow_Task	wf	wf
Workflow_Task_Templat e	wftpl	wftpl
wspcol	wspcol	wspcol
wsptbl	wsptbl	wsptbl

SQL Name to Table and Object

This table provides a cross-reference of the SQL name of each table in the database schema to its corresponding table and object names:

SQL Name (AKA)	Table	Object
acc_lvls	Access_Levels	acc_lvls
acctyp	Access_Type	acctyp
act_log	Act_Log	alg
act_type	Act_Type	aty
actbool	Active_Boolean_Table	actbool
actrbool	Active_Reverse_Boolean_Tabl e	actrbool
admin_tree	admin_tree	ADMIN_TREE
am_map	Am_Asset_Map	am_asset_map
anima	Animator	ANI
arcpur_hist	Archive_Purge_History	arcpur_hist
arcpur_rule	Archive_Purge_Rule	arcpur_rule
atn	Attribute_Name	
atomic_cond	Atomic_Condition	atomic_cond
att_evt	Attached_Events	atev

SQL Name (AKA)	Table	Object
attached_sla	Attached_SLA	attached_sla
attmnt	Attachment	attmnt
attmnt_folder	attmnt_folder	attmnt_folder
attmnt_lrel	Attachment_Lrel	attmnt_lrel
atyp_asc	Act_Type_Assoc	act_type_assoc
audit_log	Audit_Log	audlog
bhvtpl	Behavior_Template	bhvtpl
bool_tab	Boolean_Table	bool
bpwshft	Bop_Workshift	wrkshft
BU_TRANS	BU_TRANS	BU_TRANS
buscls	Business_Management_Class	bmcls
buslrel	Business_Management_Repos itory_Lrel	bmlrel
busmgt	Business_Management	bmhier
busrep	Business_Management_Repos itory	bmrep
busstat	Business_Management_Status	bms
ca_asset_type	ca_asset_type	
ca_company	ca_company	ca_cmpny
ca_company_type	ca_company_type	vpt
ca_contact	ca_contact	cnt
ca_contact_type	ca_contact_type	ctp
ca_country	ca_country	country
ca_job_function	ca_job_function	job_func
ca_job_title	ca_job_title	position
ca_location	ca_location	loc
ca_model_def	ca_model_def	mfrmod
ca_organization	ca_organization	org
ca_owned_resource	ca_owned_resource	nr
ca_resource_class	ca_resource_class	grc

SQL Name (AKA)	Table	Object
ca_resource_cost_center	ca_resource_cost_center	cost_cntr
ca_resource_department	ca_resource_department	dept
ca_resource_family	ca_resource_family	nrf
ca_resource_gl_code	ca_resource_gl_code	gl_code
ca_resource_operating_system	ca_resource_operating_system	opsys
ca_resource_status	ca_resource_status	rss
ca_schema_info	ca_schema_info	
ca_site	ca_site	site
ca_state_province	ca_state_province	state
call_req	Call_Req	cr
ccat_grp	Chgcat_Group	chgcat_grp
ccat_loc	Chgcat_Loc	chgcat_loc
ccat_wrkshft	Chgcat_Workshift	chgcat_workshift
chg	Change_Request	chg
chg_template	Chg_Template	chg_tpl
chgalg	Change_Act_Log	chgalg
chgcat	Change_Category	chgcat
chgstat	Change_Status	chgstat
CI_ACTIONS	CI_ACTIONS	CI_ACTIONS
CI_ACTIONS_ALTERATE	CI_ACTIONS_ALTERNATE	CI_ACTIONS_ALTERATE
CI_BOOKMARKS	CI_BOOKMARKS	CI_BOOKMARKS
CI_DOC_LINKS	CI_DOC_LINKS	CI_DOC_LINKS
CI_DOC_TEMPLATES	CI_DOC_TEMPLATES	CI_DOC_TEMPLATES
CI_DOC_TYPES	CI_DOC_TYPES	CI_DOC_TYPES
CI_PRIORITIES	CI_PRIORITIES	CI_PRIORITIES
CI_STATUSES	CI_STATUSES	CI_STATUSES
CI_WF_TEMPLATES	CI_WF_TEMPLATES	CI_WF_TEMPLATES

SQL Name (AKA)	Table	Object
cn	Column_Name	
cnote	Note_Board	cnote
cr_prp	Req_Property	cr_prp
cr_prptpl	Req_Property_Template	cr_prptpl
cr_stat	Cr_Status	crs
cr_template	Cr_Template	cr_tpl
crctmr	Cr_Call_Timers	ctimer
crsol	Call_Solution	crsol
crsq	Cr_Stored_Questions	crsq
crt	Call_Req_Type	crt
ct_mth	Contact_Method	cmth
ctab	Controlled_Table	ctab
D_PAINTER	D_PAINTER	
dcon	Domain_Constraint	dcon
dcon_typ	Domain_Constraint_Type	dcon_typ
dlgtsrv	Delegation_Server	dlgtsrv
dmn	Domain	dmn
doc_rep	Document_Repository	doc_rep
EBR_ACRONYMS	EBR_ACRONYMS	EBR_ACRONYMS
EBR_FULLTEXT	EBR_FULLTEXT	EBR_FULLTEXT
EBR_FULLTEXT_AD M	EBR_FULLTEXT_ADMIN	EBR_FULLTEXT_ADMIN
EBR_FULLTEXT_SD	EBR_FULLTEXT_SD	EBR_FULLTEXT_SD
EBR_FULLTEXT_SD_ ADM	EBR_FULLTEXT_SD_ADMIN	EBR_FULLTEXT_SD_ADMIN
EBR_INDEXING_QU EUE	EBR_INDEXING_QUEUE	EBR_INDEXING_QUEUE
EBR_KEYWORDS	EBR_KEYWORDS	EBR_KEYWORDS
EBR_LOG	EBR_LOG	EBR_LOG
EBR_METRICS	EBR_METRICS	EBR_METRICS
EBR_NOISE_WORDS	EBR_NOISE_WORDS	EBR_NOISE_WORDS

SQL Name (AKA)	Table	Object
EBR_PATTERNS	EBR_PATTERNS	EBR_PATTERNS
EBR_PREFIXES	EBR_PREFIXES	EBR_PREFIXES
EBR_PROPERTIES	EBR_PROPERTIES	EBR_PROPERTIES
EBR_SUBSTITS	EBR_SUBSTITS	EBR_SUBSTITS
EBR_SUFFIXES	EBR_SUFFIXES	EBR_SUFFIXES
EBR_SYNONYMS	EBR_SYNONYMS	EBR_SYNONYMS
EBR_SYNONYMSADM	EBR_SYNONYMSADM	EBR_SYNONYMSADM
ES_CONSTANTS	ES_CONSTANTS	ES_CONSTANTS
ES_NODES	ES_NODES	ES_NODES
ES_RESPONSES	ES_RESPONSES	ES_RESPONSES
ES_SESSIONS	ES_SESSIONS	ES_SESSIONS
event_log	event_log	event_log
event_type	event_type	event_type
evt	Events	evt
evt_dly	Event_Delay	evtdly
evtdlytp	Event_Delay_Type	evtdlytp
ext_appl	ext_appl	
frmgrp	Form_Group	fmgrp
g_chg_ext	Global_Change_Extension	g_chg_ext
g_chg_queue	Global_Change_Queue	g_chg_queue
g_contact	Global_Contact	g_cnt
g_iss_ext	Global_Issue_Extension	g_iss_ext
g_iss_queue	Global_Issue_Queue	g_iss_queue
g_loc	Global_Location	g_loc
g_org	Global_Organization	g_org
g_product	Global_Product	g_prod
g_queue_names	Global_Queue_Names	g_qname
g_req_ext	Global_Request_Extension	g_cr_ext
g_req_queue	Global_Request_Queue	g_cr_queue
g_srvr	Global_Servers	g_srvrs

SQL Name (AKA)	Table	Object
g_tbl_map	Global_Table_Map	g_tblmap
g_tbl_rule	Global_Table_Rule	g_tblrule
grp_loc	Group_Loc	grp_loc
grpmem	Group_Member	grpmem
hier	Asset_Assignment	hier
icat_grp	Isscat_Group	isscat_grp
icat_loc	Isscat_Loc	isscat_loc
icat_wrkshft	Isscat_Workshift	isscat_workshift
impact	Impact	imp
INDEX_DOC_LINKS	INDEX_DOC_LINKS	INDEX_DOC_LINKS
interface	Interface	intfc
iss_template	Iss_Template	iss_tpl
issalg	Issue_Act_Log	issalg
isscat	Issue_Category	isscat
issprp	Issue_Property	iss_prp
issstat	Issue_Status	issstat
issue	Issue	iss
isswf	Issue_Workflow_Task	iss_wf
kc	Key_Control	
KD_ATTMNT	KD_ATTMNT	KD_ATTMNT
kdlinks	kdlinks	kdlinks
km_kword	Knowledge_Keywords	kwd
km_lrel	Knowledge_Lrel_Table	kmlrel
KT_REPORT_CARD	KT_REPORT_CARD	KT_REPORT_CARD
LONG_TEXTS	LONG_TEXTS	LONG_TEXTS
lrel	Lrel_Table	lrel1
managed_survey	Managed_Survey	mgs
mgsalg	Mgs_Act_Log	mgsalg
mgsstat	Mgs_Status	mgsstat
not_log	Notify_Log_Header	lr

SQL Name (AKA)	Table	Object
not_que	Queued_Notify	notque
NOTIFICATION	NOTIFICATION	NOTIFICATION
nottrn	Transition_Points	
noturg	Notification_Urgency	noturg
nr_com	NR_Comment	nr_com
ntfl	Notify_Object_Attr	ntfl
O_COMMENTS	O_COMMENTS	O_COMMENTS
O_EVENTS	O_EVENTS	O_EVENTS
O_INDEXES	O_INDEXES	KCAT
options	Options	options
P_GROUPS	P_GROUPS	P_GROUPS
pcat_grp	Pcat_Group	pcat_grp
pcat_loc	Pcat_Loc	pcat_loc
pcat_wrkshft	Pcat_Workshift	pcat_workshift
perscon	Person_Contacting	perscnt
pri	Priority	pri
prob_ctg	Prob_Category	pcat
product	Product	prod
prp	Property	prp
prptpl	Property_Template	prptpl
quick_tpl_types	Quick_Template_Types	quick_tpl_types
rbooltab	Reverse_Boolean_Table	rev_bool
rem_ref	Remote_Ref	rrf
repmeth	Reporting_Method	rptmeth
response	Response	response
rootcause	Rootcause	rc
rptmth	Rpt_Meth	rptm
sapolicy	SA_Policy	sapolicy
saprobtyp	SA_Prob_Type	saprobtyp
sdsc_map	SLA_Contract_Map	sdsc_map

SQL Name (AKA)	Table	Object
seqctl	Sequence_Control	seq
session_log	session_log	session_log
session_type	session_type	session_type
sevrty	Severity	sev
SHOW_OBJ	SHOW_OBJ	SHOW_OBJ
SKELETONS	SKELETONS	KD
slatpl	SLA_Template	slatpl
splmac	Spell_Macro	macro
splmactp	Spell_Macro_Type	macro_type
sql_tab	SQL_Script	
srv_desc	Service_Desc	sdsc
srvr_aliases	Server_Aliases	srvr_aliases
srvr_zones	Server_Zones	srvr_zones
survey	Survey	survey
survey_answer	Survey_Answer	svy_ans
survey_atpl	Survey_Answer_Template	svy_atpl
survey_qtpl	Survey_Question_Template	svy_qtpl
survey_question	Survey_Question	svy_ques
survey_statistics	Survey_Stats	svystat
survey_tpl	Survey_Template	svy_tpl
survey_tracking	Survey_Tracking	svytrk
svc_contract	Service_Contract	svc_contract
tn	Table_Name	
toc	Type_Of_Contact	typecnt
tskstat	Task_Status	tskstat
tskty	Task_Type	tskty
tspan	Timespan	tspan
tz	Timezone	tz
urgncy	Urgency	urg
usp_contact	usp_contact	cnt

SQL Name (AKA)	Table	Object
usp_organization	usp_organization	org
usp_owned_resource	usp_owned_resource	nr
USP_PREFERENCES	USP_PREFERENCES	USP_PREFERENCES
USP_PROPERTIES	USP_PROPERTIES	USP_PROPERTIES
usq	User_Query	usq
wf	Workflow_Task	wf
wftpl	Workflow_Task_Template	wftpl
wftpl_grp	Wftpl_Group	wftpl_grp
wspcol	wspcol	wspcol
wsptbl	wsptbl	wsptbl
xent_map	External_Entity_Map	ext_entity_map

Object to Table and SQL Name

This table provides a cross-reference of each object to its corresponding table and SQL name in the database schema:

Object	Table	SQL Name (AKA)
acc_lvls	Access_Levels	acc_lvls
acctyp	Access_Type	acctyp
act_type_assoc	Act_Type_Assoc	atyp_asc
actbool	Active_Boolean_Table	actbool
actrbool	Active_Reverse_Boolean_Table	actrbool
ADMIN_TREE	admin_tree	admin_tree
alg	Act_Log	act_log
am_asset_map	Am_Asset_Map	am_map
ANI	Animator	anima
arcpur_hist	Archive_Purge_History	arcpur_hist
arcpur_rule	Archive_Purge_Rule	arcpur_rule
atev	Attached_Events	att_evt

Object	Table	SQL Name (AKA)
atomic_cond	Atomic_Condition	atomic_cond
attached_sla	Attached_SLA	attached_sla
attmnt	Attachment	attmnt
attmnt_folder	attmnt_folder	attmnt_folder
attmnt_lrel	Attachment_Lrel	attmnt_lrel
aty	Act_Type	act_type
audlog	Audit_Log	audit_log
bhvtpl	Behavior_Template	bhvtpl
bmcls	Business_Management_Class	buscls
bmhier	Business_Management	busmgt
bmlrel	Business_Management_Repository_Lrel	buslrel
bmrep	Business_Management_Repository	busrep
bms	Business_Management_Status	busstat
bool	Boolean_Table	bool_tab
BU_TRANS	BU_TRANS	BU_TRANS
ca_cmpny	ca_company	ca_company
chg	Change_Request	chg
chg_tpl	Chg_Template	chg_template
chgalg	Change_Act_Log	chgalg
chgcat	Change_Category	chgcat
chgcat_grp	Chgcat_Group	ccat_grp
chgcat_loc	Chgcat_Loc	ccat_loc
chgcatt_workshif	Chgcatt_Workshift	ccatt_wrkshft
chgstat	Change_Status	chgstat
CI_ACTIONS	CI_ACTIONS	CI_ACTIONS
CI_ACTIONS_AL	CI_ACTIONS_ALTERNATE	CI_ACTIONS_ALTERNATE
TERNATE		
CI_BOOKMARK_S	CI_BOOKMARKS	CI_BOOKMARKS

Object	Table	SQL Name (AKA)
CI_DOC_LINKS	CI_DOC_LINKS	CI_DOC_LINKS
CI_DOC_TEMPLATES	CI_DOC_TEMPLATES	CI_DOC_TEMPLATES
CI_DOC_TYPES	CI_DOC_TYPES	CI_DOC_TYPES
CI_PRIORITIES	CI_PRIORITIES	CI_PRIORITIES
CI_STATUSES	CI_STATUSES	CI_STATUSES
CI_WF_TEMPLATES	CI_WF_TEMPLATES	CI_WF_TEMPLATES
cmth	Contact_Method	ct_mth
cnote	Note_Board	cnote
cnt	ca_contact	ca_contact
cnt	usp_contact	usp_contact
cost_cntr	ca_resource_cost_center	ca_resource_cost_center
country	ca_country	ca_country
cr	Call_Req	call_req
cr_prp	Req_Property	cr_prp
cr_prptpl	Req_Property_Template	cr_prptpl
cr_tpl	Cr_Template	cr_template
crs	Cr_Status	cr_stat
crsol	Call_Solution	crsol
crsq	Cr_Stored_Questions	crsq
crt	Call_Req_Type	crt
ctab	Controlled_Table	ctab
ctimer	Cr_Call_Timers	crctmr
ctp	ca_contact_type	ca_contact_type
dcon	Domain_Constraint	dcon
dcon_typ	Domain_Constraint_Type	dcon_typ
dept	ca_resource_department	ca_resource_department
dlgsvr	Delegation_Server	dlgtsrv
dmn	Domain	dmn
doc_rep	Document_Repository	doc_rep

Object	Table	SQL Name (AKA)
EBR_ACRONYM_S	EBR_ACRONYMS	EBR_ACRONYMS
EBR_FULLTEXT	EBR_FULLTEXT	EBR_FULLTEXT
EBR_FULLTEXT_ADMIN	EBR_FULLTEXT_ADMIN	EBR_FULLTEXT_ADMIN
EBR_FULLTEXT_SD	EBR_FULLTEXT_SD	EBR_FULLTEXT_SD
EBR_FULLTEXT_SD_ADMIN	EBR_FULLTEXT_SD_ADMIN	EBR_FULLTEXT_SD_ADMIN
EBR_INDEXING_QUEUE	EBR_INDEXING_QUEUE	EBR_INDEXING_QUEUE
EBR_KEYWORDS	EBR_KEYWORDS	EBR_KEYWORDS
EBR_LOG	EBR_LOG	EBR_LOG
EBR_METRICS	EBR_METRICS	EBR_METRICS
EBR_NOISE_WORDS	EBR_NOISE_WORDS	EBR_NOISE_WORDS
EBR_NOISE_WORDS_ORDS	EBR_NOISE_WORDS	EBR_NOISE_WORDS
EBR_PATTERNS	EBR_PATTERNS	EBR_PATTERNS
EBR_PREFIXES	EBR_PREFIXES	EBR_PREFIXES
EBR_PROPERTIES	EBR_PROPERTIES	EBR_PROPERTIES
EBR_SUBSTITUTIONS	EBR_SUBSTITUTIONS	EBR_SUBSTITUTIONS
EBR_SUFFIXES	EBR_SUFFIXES	EBR_SUFFIXES
EBR_SYNONYMS	EBR_SYNONYMS	EBR_SYNONYMS
EBR_SYNONYMS_ADMIN	EBR_SYNONYMS_ADMIN	EBR_SYNONYMS_ADMIN
ES_CONSTANTS	ES_CONSTANTS	ES_CONSTANTS
ES_NODES	ES_NODES	ES_NODES
ES_RESPONSES	ES_RESPONSES	ES_RESPONSES
ES_SESSIONS	ES_SESSIONS	ES_SESSIONS
event_log	event_log	event_log
event_type	event_type	event_type
evt	Events	evt

Object	Table	SQL Name (AKA)
evtdly	Event_Delay	evt_dly
evtdlytp	Event_Delay_Type	evtdlytp
ext_entity_map	External_Entity_Map	xent_map
fmgrp	Form_Group	frmgrp
g_chg_ext	Global_Change_Extension	g_chg_ext
g_chg_queue	Global_Change_Queue	g_chg_queue
g_cnt	Global_Contact	g_contact
g_cr_ext	Global_Request_Extension	g_req_ext
g_cr_queue	Global_Request_Queue	g_req_queue
g_iss_ext	Global_Issue_Extension	g_iss_ext
g_iss_queue	Global_Issue_Queue	g_iss_queue
g_loc	Global_Location	g_loc
g_org	Global_Organization	g_org
g_prod	Global_Product	g_product
g_qname	Global_Queue_Names	g_queue_names
g_srvers	Global_Servers	g_srvr
g_tblmap	Global_Table_Map	g_tbl_map
g_tblrule	Global_Table_Rule	g_tbl_rule
gl_code	ca_resource_gl_code	ca_resource_gl_code
grc	ca_resource_class	ca_resource_class
grp_loc	Group_Loc	grp_loc
grpmem	Group_Member	grpmem
hier	Asset_Assignment	hier
imp	Impact	impact
INDEX_DOC_LI NKS	INDEX_DOC_LINKS	INDEX_DOC_LINKS
intfc	Interface	interface
iss	Issue	issue
iss_prp	Issue_Property	issprp
iss_tpl	Iss_Template	iss_template
iss_wf	Issue_Workflow_Task	isswf

Object	Table	SQL Name (AKA)
issalg	Issue_Act_Log	issalg
isscat	Issue_Category	isscat
isscat_grp	Isscat_Group	icat_grp
isscat_loc	Isscat_Loc	icat_loc
isscat_workshift	Isscat_Workshift	icat_wrkshft
issstat	Issue_Status	issstat
job_func	ca_job_function	ca_job_function
KCAT	O_INDEXES	O_INDEXES
KD	SKELETONS	SKELETONS
KD_ATTMNT	KD_ATTMNT	KD_ATTMNT
kdlinks	kdlinks	kdlinks
kmlrel	Knowledge_Lrel_Table	km_lrel
KT_REPORT_CA RD	KT_REPORT_CARD	KT_REPORT_CARD
kwd	Knowledge_Keywords	km_kwrd
loc	ca_location	ca_location
LONG_TEXTS	LONG_TEXTS	LONG_TEXTS
lr	Notify_Log_Header	not_log
lrel1	Lrel_Table	lrel
macro	Spell_Macro	splmac
macro_type	Spell_Macro_Type	splmactp
mfrmod	ca_model_def	ca_model_def
mgs	Managed_Survey	managed_survey
mgsalg	Mgs_Act_Log	mgsalg
mgsstat	Mgs_Status	mgsstat
NOTIFICATION	NOTIFICATION	NOTIFICATION
notque	Queued_Notify	not_que
noturg	Notification_Urgency	noturg
nr	ca_owned_resource	ca_owned_resource
nr	usp_owned_resource	usp_owned_resource
nr_com	NR_Comment	nr_com

Object	Table	SQL Name (AKA)
nrf	ca_resource_family	ca_resource_family
ntfl	Notify_Object_Attr	ntfl
O_COMMENTS	O_COMMENTS	O_COMMENTS
O_EVENTS	O_EVENTS	O_EVENTS
opsys	ca_resource_operating_system	ca_resource_operating_system
options	Options	options
org	ca_organization	ca_organization
org	usp_organization	usp_organization
P_GROUPS	P_GROUPS	P_GROUPS
pcat	Prob_Category	prob_ctg
pcat_grp	Pcat_Group	pcat_grp
pcat_loc	Pcat_Loc	pcat_loc
pcat_workshift	Pcat_Workshift	pcat_wrkshft
perscnt	Person_Contacting	personcon
position	ca_job_title	ca_job_title
pri	Priority	pri
prod	Product	product
prp	Property	prp
prptpl	Property_Template	prptpl
quick_tpl_types	Quick_Template_Types	quick_tpl_types
rc	Rootcause	rootcause
response	Response	response
rev_bool	Reverse_Boolean_Table	rbooltab
rptm	Rpt_Meth	rptmth
rptmeth	Reporting_Method	repmeth
rrf	Remote_Ref	rem_ref
rss	ca_resource_status	ca_resource_status
sapolicy	SA_Policy	sapolicy
saprobtyp	SA_Prob_Type	saprobtyp
sdsc	Service_Desc	srv_desc

Object	Table	SQL Name (AKA)
sdsc_map	SLA_Contract_Map	sdsc_map
seq	Sequence_Control	seqctl
session_log	session_log	session_log
session_type	session_type	session_type
sev	Severity	sevrty
SHOW_OBJ	SHOW_OBJ	SHOW_OBJ
site	ca_site	ca_site
slatpl	SLA_Template	slatpl
srvr_aliases	Server_Aliases	srvr_aliases
srvr_zones	Server_Zones	srvr_zones
state	ca_state_province	ca_state_province
survey	Survey	survey
svc_contract	Service_Contract	svc_contract
svy_ans	Survey_Answer	survey_answer
svy_atpl	Survey_Answer_Template	survey_atpl
svy_qtpl	Survey_Question_Template	survey_qtpl
svy_ques	Survey_Question	survey_question
svy_tpl	Survey_Template	survey_tpl
svystat	Survey_Stats	survey_statistics
svytrk	Survey_Tracking	survey_tracking
tskstat	Task_Status	tskstat
tskty	Task_Type	tskty
tspan	Timespan	tspan
typecnt	Type_Of_Contact	toc
tz	Timezone	tz
urg	Urgency	urgnc
USP_PREFEREN CES	USP_PREFERENCES	USP_PREFERENCES
USP_PROPERTI ES	USP_PROPERTIES	USP_PROPERTIES
usq	User_Query	usq

Object	Table	SQL Name (AKA)
vpt	ca_company_type	ca_company_type
wf	Workflow_Task	wf
wftpl	Workflow_Task_Template	wftpl
wftpl_grp	Wftpl_Group	wftpl_grp
wrkshft	Bop_Workshift	bpwshft
wspcol	wspcol	wspcol
wsptbl	wsptbl	wsptbl
	Attribute_Name	atn
	ca_asset_type	ca_asset_type
	ca_schema_info	ca_schema_info
	Column_Name	cn
D_PAINTER	D_PAINTER	
ext_appl	ext_appl	
Key_Control	KC	
Transition_Points	nottrn	
SQL_Script	sql_tab	
Table_Name	TN	

Appendix D: Schema Files Syntax

The Unicenter Service Desk database schema is defined in multiple .sch files in the \$NX_ROOT/site (UNIX) or *installation-directory*/site (Windows) directory. During configuration, these .sch files along with any customized .sch files you may create are merged together into a single file called \$NX_ROOT/site/ddict.sch (UNIX) or *installation-directory*/site/ddict.sch (Windows).

Please review the chapter Customizing the Schema Using the Web Screen Painter in this guide prior to making any changes to these files.

Note: Do not modify any .sch files in the \$NX_ROOT/site (UNIX) or *installation-directory*/site (Windows) directory. Any changes you make to these files will be lost when migration to a new release or when certain patches are applied. If you want to make schema changes, you must create a file in the \$NX_ROOT/site/mods (UNIX) or *installation-directory*/site/mods (Windows) directory with the file suffix .sch. Then add your changes to your .sch file, and your changes will then be merged with delivered schema during configuration. This is the only way to preserve your modifications when you upgrade to a new release.

TABLE Statement

Defines the logical tables in the Unicenter Service Desk database schema and the logical columns (fields) in those tables. These logical tables and columns are then mapped to the physical tables and columns used by your database management system in a mapping statement that follows the TABLE statement.

Note: If you define a new table, you must define a mapping statement for that table. The Mapping Statement (see page 702) is illustrated at the end of this appendix, followed by an example that combines the TABLE, TABLE_INFO, and mapping statements.

Syntax

```
TABLE table_name { field value_type field_attributes; [...] }
```

Arguments

TABLE

Introduces the TABLE statement. Must be uppercase. You must have one TABLE statement for each logical table in the schema.

table_name

The name of the database table, for example, Call_Req. If adding a database table, you can specify any name beginning with a lowercase letter z. (This avoids possible conflict with existing and future Unicenter Service Desk table names.) If changing an existing table, find the table in one of the .sch files and use the same name.

field

The name of a logical column in the table, for example, id or desc. You must identify each column by name. If adding a table or adding a column to an existing table, you can specify any name beginning with a lowercase letter z; however, field names must not end with the characters “_f.” (This avoids possible conflict with existing and future Unicenter Service Desk column names.) If changing an existing column, find the column in one of the .sch files and use the same name.

value_type

The field's data type. Valid values are:

Value	Description
STRING <i>nn</i>	A string that is <i>nn</i> characters long.
INTEGER	A 32-bit number.

Value	Description
LOCAL_TIME	The number of seconds since January 1, 1970. Unicenter Service Desk automatically reformats this data type to the designated date format, for example: <i>mm/dd/yy hh:mm:ss</i> .
DURATION	A period of time, measured in seconds.
REAL	A floating point number
UUID	A 16 byte binary value.

field_attributes

A description of the field. Valid values are:

Value	Description
KEY	Identifies this field as the primary key to be used for identifying records to be updated with pdm_load. This is used if the default primary key, id, is not specified. Must be specified if the field is the primary key in the table.
NOT_NULL	Indicates that the field must contain a value. Must be specified if the field is the primary key in the table. Optional if the field is not the primary key.
REF other_table_name	Indicates that the field references another table. Optional whether the field is the primary key or not.
S_KEY	Optionally identifies this field as the secondary key to be used for identifying records to be updated with pdm_load.
UNIQUE	Indicates that the values in the field must be unique. Must be specified if the field is the primary key in the table. Optional if the field is not the primary key.

Macros are synonyms that will be converted during configuration to the value the macro represents. You can use macros for either data types or attributes. If you wish to use macros, you must add in #include statement to include the file that defines the macro including the path name (usually relative to your schema file). The include statement must be defined prior to using the macro. Example of an include statement:

```
#include ".../schema.mac"
```

The following are some of the macros defined in .mac files located in the \$ NX_ROOT/site (UNIX) or *installation-directory*/site (Windows) directory.

Data Type	Equivalent
nn	NOT_NULL
uniq	UNIQUE NOT_NULL
ADDR_LINE	STRING 30
EMAILADDR	STRING 120
ENT_DESC	STRING 40
ENT_NAME	STRING 30
OSI_NAME	STRING 80
OSI_TYPE_STRING	STRING 60
USERID	STRING 85
PHONENUM	STRING 32
SYMBOL	STRING 12
HIER_SYM	STRING 60
LONG_SYM	STRING 30
COMMENT	STRING 1000
LONG_STR	STRING 500
LONG_DESC	STRING 240
BOOL	INTEGER

Examples

This TABLE statement in the database schema defines severities. The macro nn indicates that a value is required in the del field. The macro uniq indicates that values are required and must be unique:

```
#include :.../schema.mac"
TABLE Severity {
    id      INTEGER uniq KEY;    // key id
    del    INTEGER nn;           // 0=present,1=gone
    sym    SYMBOL uniq S_KEY;   // type symbol
    desc   ENT_DESC;            // non-OSI specified column
}
```

This modified TABLE statement makes the Priority field on the Request Detail window required:

```
TABLE Call_Req {priority INTEGER NOT_NULL;}
```

This TABLE statement adds a resolution code field to the Call_Req table. The content of the field is numeric and references the Resolution_Code table. This reference allows users to double-click the Resolution Code field on the Request Detail window to display the values in the Resolution_Code table:

```
TABLE Call_Req {zres_code INTEGER REF Resolution_Code;}
```

TABLE_INFO Statement

This instructs your database management system how to store and index data in the logical tables. The extent to which these instructions are followed depends on the database management system. If no instructions are provided, the database management system follows its own storage and indexing instructions.

Syntax

```
TABLE_INFO table_name {  
    [STORAGE storage_mtd Field ;]  
    [INDEX ndx_props field1 [field2 ...];] ...}
```

Arguments

TABLE_INFO

Introduces the TABLE_INFO statement. Must be uppercase. The TABLE_INFO statement is optional, but if specified, you can have only one TABLE_INFO for each TABLE statement, and it must follow the TABLE statement.

table_name

The name of the database table in the TABLE statement.

STORAGE *storage_mtd*

Identifies the storage method. Valid values are listed as follows, but note that some database management systems ignore these values:

Value	Description
BTREE	Indicates to use the balanced tree storage method.
HASH	Indicates to use the hash table storage method. This is valid only if the field is the primary key.
HEAP	Indicates to use the heap storage method.

field

Identifies the column that is to be stored according to the specified storage method (STORAGE *storage_mtd*). Must be specified the same way as the name of the column in the TABLE statement.

INDEX *ndx_props*

Identifies one or more properties for an index that consists of the fields specified. Valid values are:

Value	Description
SORT ASCENDING DESCENDING	Indicates whether to sort the data in the fields in ascending or descending order. Data is sorted in ascending order by default; therefore, only SORT DESCENDING need be specified.
PRIMARY	Indicates to use this index as the default sort order for the table.
CLUSTER	Identifies this as a clustering index.
UNIQUE	Indicates that values in the index must be unique.

***field1* [*field2* . . .]**

Identifies the column or columns that are to be indexed according to the specified index properties (INDEX *ndx_props*). Must be specified the same way as the name of the columns in the TABLE statement.

Examples

This TABLE_INFO statement instructs the database management system to use a hash table to store values in the id field in the Contact_Type table, and to sort the table in descending order according to the values in the sym field. It also indicates that values must be unique:

```
TABLE_INFO Contact_Type {
    STORAGE HASH id;
    INDEX SORT DESCENDING PRIMARY UNIQUE sym;
}
```

Mapping Statement

Defines the correspondence between the logical tables and columns in the Unicenter Service Desk database schema and the physical tables and columns used by your database management system. This statement follows each TABLE statement in a.sch file. You must define it when you define a new table.

Syntax

```
p1 logical_table_name -> CURR_PROV physical_table_name  
  [{logical_field -> physical_field ...} ;  
  {}]
```

Arguments

p1

Introduces the mapping statement. Must be specified as p1.

logical_table_name

The name of the database table in the TABLE statement, for example, zManufacturer.

CURR_PROV

A required keyword.

physical_table_name

The name of the table used by your database management system, for example, man. Short names improve performance and are required by some database management systems.

logical_field

The name of the column in the Unicenter Service Desk database schema, for example, desc. Must be the same as *field* in the TABLE statement. Omit this when the logical columns and physical columns have identical names. When omitted, the semicolon follows *physical_table_name*.

physical_field

The name of the column used by your database management system, for example, nx_desc. Omit this when the logical columns and physical columns have identical names. When omitted, the semicolon follows *physical_table_name*.

Examples

This example illustrates how TABLE, mapping (p1), and TABLE_INFO statements define a zManufacturer table:

```
TABLE zManufacturer {
```

```
    id      INTEGER  uniq KEY;          // key id
    del     INTEGER  nn;                // 0=present,1=gone
    sym     HIER_SYM uniq S_KEY;       // manufacturer name
    desc    ENT_DESC;                 // manufacturer description
}

p1 zManufacturer -> CURR_PROV man  // maps logical table "zManufacturer"
{
    desc -> nx_desc;              // maps logical column "desc"
}                                    // to physical column "nx_desc"

TABLE_INFO zManufacturer {
    STORAGE HASH id;
    INDEX SORT ASCENDING PRIMARY UNIQUE sym;
}
```


Appendix E: Object Definition Syntax

Many of the components of Unicenter Service Desk consist of business objects. These objects are defined in a metalanguage named Majic. You can use Majic statements to create new objects and modify existing objects, thus customizing these objects to meet your needs.

Directories

The Majic files are organized in two directories:

Directory	Description
bopcfg/majic (UNIX) or bopcfg\majic (Windows)	Contains the .maj files that have been used to create windows defined in the database. These files should not be changed because changes will be overwritten by new releases of Unicenter Service Desk.
site/mods/majic (UNIX) or site\mods\majic (Windows)	Contains the .mod files you use to customize windows.

Types of Statements

The following Majic statements have been used in the procedures in the chapters "Using Screen Painter" and "Customizing the Database Schema" chapters:

Statement	Description
OBJECT	Defines a business object
MODIFY	Changes existing object attributes
MODIFY FACTORY	Changes existing factories

These Majic statements are described in detail in this appendix.

OBJECT Statement

Defines a business object.

Syntax

```
OBJECT obj_name {  
    [ATTRIBUTES [table_name]{  
        att_name [field_name] value_type [access_type[status_type]] [DISPLAY_NAME  
string][{  
            [ON_NEW DEFAULT|SET value|NOW ;]  
            [ON_CI DEFAULT|SET value|NOW ;]  
            [ON_DB_INIT DEFAULT|SET value|NOW ;]} ;]};]  
  
    [FACTORY [fac_name]{  
        [REL_ATTR name ;]  
        [COMMON_NAME name ;]  
        [DISPLAY_NAME name ;]  
        [FUNCTION_GROUP name ;]  
        [STANDARD_LISTS {  
            [SORT_BY index_att ;]  
            [FETCH fetch_att ;]  
            [WHERE string ;]  
            [MLIST ON|OFF;]  
            [RLIST ON|OFF;] } ;]};]  
};
```

Arguments

obj_name

The object's name (for example, cnt for contact or cr for request).

Optional Statements

Either ATTRIBUTES or FACTORY must be specified. Both can be specified.

ATTRIBUTES [*table_name*] { }

Defines the properties of the object. Most attributes map to a field (column) in a database table. The ATTRIBUTES Optional Statement (see page 708) describes its syntax.

FACTORY [*fac_name*] { }

Defines access to the object, like its relation attribute, a common name, the security group that can access it, the type of lists produced, and how those lists can be sorted. The FACTORY Optional Statement (see page 712) describes its syntax.

Example

This example defines an object named ctp. The ATTRIBUTES statement defines attributes named sym, delete_flag, and description whose values are stored in the Contact_Type table in the database. The FACTORY statement creates a master list of objects, sorted by values in the field that corresponds to the sym attribute, and specifies that the id attribute will represent ctp when it is referenced by an SREL:

```
OBJECT ctp {
    ATTRIBUTES Contact_Type {
        sym          STRING REQUIRED ;
        delete_flag del   INTEGER {
            ON_NEW DEFAULT 0 ;
        } ;
        description desc  STRING ;
    } ;
    FACTORY {
        STANDARD_LISTS {SORT_BY "sym"} ;
        REL_ATTR id ;
    };
}
```

ATTRIBUTES Optional Statement

The optional statement on the OBJECT statement that defines the properties of the object.

Syntax

```
ATTRIBUTES [table_name]{
    att_name [field_name] value_type [access_type[status_type][DISPLAY_NAME
string]]{
        [ON_NEW DEFAULT|SET value|NOW ;]
        [ON_CI DEFAULT|SET value|NOW ;]
        [ON_DB_INIT DEFAULT|SET value|NOW ;];};}
```

Arguments

table_name

The name of the table in the database that stores the values associated with the attributes in the object. If the table name is not specified, the *obj_name* in the OBJECT statement is used.

att_name

The name of the attribute. Each attribute usually maps to a field (column) in the database table.

field_name

The name of the field in the database table or LOCAL if the attribute does not map to a field or DERIVED (derived-expr) if the attribute is derived from other attributes. If neither LOCAL, DERIVED nor a field name is specified, the name of the field is assumed to be the same as the name of the attribute.

A variable declared as DERIVED is constructed only when its value is retrieved. The operand of DERIVED contains a list of attribute names and string constants separated by spaces. All attributes in a derived value must be simple values (that is, they cannot be xRELS), and should be declared prior to the derived variable. The derived attribute's value is the concatenation of the values of its constituent values.

String constants within a derived expression may contain references to environment variables in the one of the forms:

```
 ${var}
 ${var#pattern}
 ${var#pattern#replacement}
```

Such specifications are replaced with the value of the environment variable at domsrvr startup time. The #pattern operand is optional. If provided, it is treated as a regular expression, and replaced wherever it appears in the environment variable's value. The #replacement operand defaults to null if not specified. Because # is a fixed delimiter, the pattern cannot contain a # symbol. There are no restrictions on the use of derived attributes in other messages. They behave in same way as standard attributes. A hotlink for a derived attribute fires whenever any of the attributes from which it is built changes.

value_type

Identifies the data type of the attribute's value as:

- INTEGER
- DOUBLE
- STRING [*length*]
- DURATION
- UUID
- DATE
- SREL *obj2_name*
- SREL { ob2_name_name srel_name (name, name2, ...) }

If STRING is specified, the size can be specified in an integer following STRING. If no size is specified, the value in the database is used.

UUID is 16 bytes of binary data that is used as a unique identifier for certain database records.

SREL refers the attribute to another object. If SREL is specified, *obj2_name* must be specified to identify the object that the attribute refers to.

srel_name specifies a "named" SREL. Like a "simple" SREL, a "named" is a type of MAJIC OBJECT attribute that represents a single relation, which uniquely identifies a row in another table (ob2_name). A "simple" SREL attribute normally maps to the "id" field in another table, however a "named" SREL maps two or more attributes (name, name2, ...) to two or more attributes in the referenced table that uniquely identify a row in the referenced table.

access_type

Defines access to the attribute. Valid values are:

Value	Description
CONST	Cannot be changed
PRIVATE	Read-only

Value	Description
PUBLIC	Read/write access (the default)
WRITE_NEW	Can be written only when the object is created, before the object is saved

status_type

Indicates the status of the attribute as:

- REQUIRED
- NOT_REQUIRED (the default)

DISPLAY_NAME string

Specifies a string to be used in place of the attribute name in messages concerning this attribute, such as "required attribute missing"

ON Statements

Use one of these only when *value_type* is INTEGER, STRING, DATE, or SREL.

ON_NEW DEFAULT|SET *value*|NOW

Indicates to set the value of an attribute when the object is being created for the first time:

Value	Description
DEFAULT	Changes a null current value to <i>value</i> or NOW.
SET	Changes any current value to <i>value</i> or NOW.
<i>value</i>	Specifies a numeric value or a string value, depending on the data type of the attribute.
NOW	Specify this if the attribute is of type DATE; it sets the attribute to the current date and time.

In the following example, 90 is the value set as a default when the object is created:

```
ON_NEW DEFAULT 90 ;
```

ON_CI DEFAULT|SET *value*|NOW

Indicates to set the value of an attribute when the attribute is being checked into the database. See the description of each parameter for ON_NEW.

ON_DB_INIT DEFAULT|SET *value*|NOW

Indicates to set the value of an attribute when the attribute is being instantiated from the database. See the description of each parameter for ON_NEW.

Example

This example defines attributes with names like start_date whose values are stored in fields like nlh_start in the Notify_Log_Header table in the database. The field names are followed by each attribute's data type. Optional parameters define access to some of the attributes, indicate that the attribute is required, and tell when to set the value of some of the attributes to the current date and time.

For example, an attribute named last_mod is defined; its value is set to the current date and time when the attribute is checked into the database. An attribute named contact is also defined; its value is a single relation stored in database field nlh_c_addressee. The object referred to is cnt:

```
ATTRIBUTES Notify_Log_Header {
    start_date          nlh_start           DATE WRITE_NEW {ON_NEW DEFAULT NOW;} ;
    last_mod            nlh_start           DATE {ON_CI SET NOW;} ;
    msg_hdr             nlh_hdr              STRING 20 WRITE_NEW ;
    msg_text            nlh_msg              STRING WRITE_NEW ;
    msg_ack             nlh_user_ack        STRING ;
    contact             nlh_c_addressee      SREL cnt WRITE_NEW ;
    notify_method       nlh_cm_method       INTEGER WRITE_NEW ;
    activity_notify     nlh_transition      INTEGER WRITE_NEW ;
    pri_event           nlh_pri              INTEGER WRITE_NEW ;
    notify_type         nlh_type             INTEGER WRITE_NEW ;
    ack_time            nlh_ack_time        DURATION ;
    status              nlh_status           INTEGER REQUIRED ;
    end_date            nlh_end              DATE {ON_NEW DEFAULT NOW;} ;
};
```

FACTORY Optional Statement

Defines access to the object, like its relation attribute, a common name, the security group that can access it, the type of lists produced, and how those lists can be sorted. If omitted, the object is treated according to default specifications.

Syntax (FACTORY Optional Statement)

```
FACTORY [fac_name]{  
    [REL_ATTR name ;]  
    [COMMON_NAME name ;]  
    [FUNCTION_GROUP name ;]  
    [DISPLAY_NAME name ;]  
    [STANDARD_LISTS {  
        [SORT_BY index_att ;]  
        [FETCH fetch_att ;]  
        [WHERE string ;]  
        [MLIST ON|OFF;]  
        [RLIST ON|OFF;] } ;]  
};
```

Arguments (FACTORY Optional Statement)

fac_name

The name of the factory that initiates the object. Specify this only if it is different from the name of the object. For example, the cnt object has four factories: cnt, cst, agt, grp.

Optional Statements (FACTORY Optional Statement)

At least one of these optional statements must be specified:

REL_ATTR *name*

Identifies the attribute that will represent this object when it is referenced (used as an SREL) by another object. Here is an example:

```
REL_ATTR id ;
```

REL_ATTR *name*, srel_name (*attr1,attr2,...*)

Identifies the attributes that will represent this object when it is referenced (used as an "named" SREL) by another object. where

srel_name

matches the "named" SREL name

attr1

is mapped to by the first attribute in the "named" SREL attribute list.

attr2

is mapped to by the second attribute in the "named" SREL attribute list.

DISPLAY_NAME *name*

Defines an external name for the table.

```
DISPLAY_NAME "Call Request" ;
```

COMMON_NAME *name*

Defines the attribute to be displayed in drop-down lists or when the user double-clicks a field, as well as when the tag does not specify a complete attribute. In the first example, the value for sym appears on the window instead of the value for the REL_ATTR. The second example allows you to specify a tag as cr.customer instead of cr.customer.combo_name.

```
COMMON_NAME sym ;
```

```
COMMON_NAME combo_name ;
```

FUNCTION_GROUP *name*

Indicates which security access group is permitted to access the object.
For example:

```
FUNCTION_GROUP "admin" ;
```

STANDARD_LISTS { }

Creates lists of objects that are kept in a cache and can be displayed on list or select windows. The parameters determine whether the lists are master lists or restricted lists, whether the objects included in the list must meet specified conditions, how the lists can be sorted, and what additional attributes are stored. Refer STANDARD_LISTS Optional Statement (see page 714) for the description of the syntax.

Example (FACTORY Optional Statement)

This example defines access to the object. A master list is produced. It can be sorted according to the values in the object's sym and code attributes. When first displayed, it is sorted according to the values in the sym attribute by default. When referenced by another object, this object is represented by the code attribute. When displayed in a window, the value for sym appears instead of the value for code. Only users in the admin security group can accessed it:

```
FACTORY {  
    STANDARD_LISTS {SORT_BY "sym,code"} ;  
    REL_ATTR code ;  
    COMMON_NAME sym ;  
    FUNCTION_GROUP "admin" ;  
};
```

STANDARD_LISTS Optional Statement

The optional statement on the FACTORY statement that defines the object's standard lists.

Syntax (STANDARD_LISTS Optional Statement)

```
STANDARD_LISTS {  
    [SORT_BY index_att ;]  
    [FETCH fetch_att ;]  
    [WHERE string ;]  
    [MLIST ON|OFF;]  
    [RLIST ON|OFF;] } ;
```

Optional Statements (STANDARD_LISTS Optional Statement)

At least one of these optional statements must be specified:

SORT_BY *index_att*

Defines the attributes that can be used to sort the standard lists. If specified, a master list is produced. Attributes must be enclosed in quotes and separated by commas. When displayed in a list or select window, the list is sorted by the first attribute, by default. For example:

```
SORT_BY "sym, code" ;
```

FETCH *fetch_att*

Specifies additional attributes to keep in the cache, besides those used to sort the list. They must be enclosed in quotes and separated by commas. For example:

```
FETCH "description" ;
```

WHERE *string*

Specifies a condition, in SQL format and surrounded by quotes, that must be met for an object to be included in a restricted list. If specified, a restricted list is produced. This example specifies that the restricted list contain only records that were not deleted:

```
WHERE "delete_flag = 0" ;
```

MLIST ON|OFF

Indicates whether to produce a master list, which includes all objects, using one of the following values:

Value	Description
ON	Produces a master list (default if SORT_BY is specified)
OFF	Does not produce a master list (default if SORT_BY is not specified or has no value defined)

RLIST ON|OFF

Indicates whether to also produce a restricted list, which includes only the objects that meet the criteria in the WHERE clause, using one of the following values:

Value	Description
ON	Produces a restricted list (default if WHERE is specified)
OFF	Does not produce a restricted list (default if WHERE is not specified or has no value defined.)

Note: RLISTs can speed up access and display but they use memory. They are usually used in select windows.

Important! MLIST OFF must be specified if you specify RLIST OFF.

Example (STANDARD_LISTS Optional Statement)

This example provides both a master list and a restricted list. Both lists contain the values defined for the sym, code, and description attributes. The records in the list can be sorted according to the values in the sym and code attributes. The restricted list contains only records that were not deleted:

```
STANDARD_LISTS {  
    SORT_BY "sym,code" ;  
    FETCH "description" ;  
    WHERE "delete_flag = 0" ;  
};
```

MODIFY Statement

Changes the way attributes are defined on OBJECT statements. MODIFY statements are read after OBJECT statements.

Syntax

```
MODIFY obj_name att_name [status_type;]  
  [ON_NEW DEFAULT|SET value|NOW ;]|  
  [ON_CI DEFAULT|SET value|NOW ;]|  
  [ON_DB_INIT DEFAULT|SET value|NOW ;]
```

Arguments

obj_name

Identifies the object whose attribute is being modified.

att_name

Identifies the attribute being modified.

status_type

Indicates whether the attribute is required. Attributes that were not originally required can be changed to REQUIRED, but not the other way around. Therefore, the only valid value is REQUIRED. Specify REQUIRED or one of the ON statements.

ON Statements

See ON Statements for a description of these statements.

Example

This example changes the salary attribute in the emp object so that it is now a required attribute:

```
MODIFY emp salary REQUIRED;
```

MODIFY FACTORY Statement

Changes the way factories are defined on OBJECT statements. MODIFY statements are read after OBJECT statements.

Syntax

```
MODIFY FACTORY fac_name {  
    [FUNCTION_GROUP name ;]  
    [DISPLAY_NAME name ;]  
    [STANDARD_LISTS {  
        [SORT_BY index_att ;]  
        [FETCH fetch_att ;]  
        [WHERE string ;]  
        [MLIST ON|OFF;]  
  
        [RLIST ON|OFF;] } ;] };
```

Arguments

fac_name

Identifies the factory, if included on the original OBJECT statement.

Optional Statements

At least one of these optional statements must be specified.

FUNCTION_GROUP *name*

Indicates which security access groups are permitted to access the object.
For example:

```
FUNCTION_GROUP "admin" ;
```

DISPLAY_NAME *name*

Defines an external name for the table.

```
DISPLAY_NAME "Call Request" ;
```

STANDARD_LISTS

Creates lists of objects that are kept in a cache. The parameters determine whether the lists are master lists or restricted lists, whether the objects included in the list must meet specified conditions, and how the lists can be sorted. Refer STANDARD_LISTS Optional Statements (see page 714) for a description of the syntax.

Index

A

activity
 log attributes • 77
 notification messages • 73
adding
 fields to reports • 37
 notification methods • 34
 reports • 37
alias keyword • 56
Attributes statement • 736

B

beeper notification • 27
bhvtpl object • 496
block statements
 BLOCK • 55
 detailed description • 55
 overview • 40
bool object • 501
bopcfg directory • 211

C

chg object • 504
chg_tplt • 507
chgalg object • 508
chgcat object • 510
chgstat object • 509
close_entry object • 511
cnote object • 520
cnt object • 521
COMMON_NAME parameter • 741
counter fields • 65
cr object • 527
cr_prp object • 529
cr_prptpl object • 530
Cr_Stored_Queries table • 65
cr_tpl object • 531
crs object • 532
crsol object • 533
crsq object • 534
crt object • 535
ctab object • 536
ctimer object • 537
ctp object • 538

customizing
 database schema • 211
 forms • 215
 labels • 67
 notification
 messages • 73
 methods • 25
 object schema • 211
 queries • 65
 reports • 37
 steps
 for notification methods • 33
 for reports • 37
 the main menu • 65
 the scoreboard • 65
 WHERE clauses • 67

D

data
 fields • 41
database
 relating to schema • 730
 schema customization • 211
dcon object • 539
ddict.sch file • 211
DEFAULT form group • 213, 215
dialog, displaying • 48
dlgsrvr • 541
dmn object • 542
doc_rep object • 543

E

email notification • 27
environment variables in notification method •
 27
evt object • 563
evtdly object • 565
evtdlytp object • 566
examples
 customizing

-
- notification messages • 78, 79
notification methods • 34
reports • 44
defining
 objects • 742
 scoreboard queries • 67
generating lists of objects • 744
of queries
 in labels • 67
 in WHERE clauses • 67
ext_entity_map object • 567
External_Entity_Map table • 345
- F**
- FACTORY statement • 740
fax notification • 27
FETCH parameter • 743
files
 .maj • 211
 .mod • 733
 .sch • 211
 ddict.sch • 211
 index.sch • 723
 schema.sch • 723
footer layout statement • 57
form
 customizing definitions • 212
 editor • 209
 groups • 215
 naming conventions • 214
Form_Group table • 346
FUNCTION_GROUP parameter • 741
functions
 report • 52
- G**
- generating custom reports • 37
Geo_Coord_Type table • 347
grc object • 583
grp_loc object • 584
grpmem object • 585
- H**
- header layout statement • 58
Header2 layout statement • 59
hier object • 586
- I**
- imp object • 587
Impact table • 363
Index.sch file • 723
Interface table • 364
intfc object • 588
iss object • 589
iss_prp object • 593
ISS_Template table • 365
iss_tpl object • 594
iss_wf object • 595
isscat object • 597
issstat object • 601
Issue Management
 architecture • 210
 form definitions • 212
 viewer • 210
Issue table • 368
Issue_Act_Log table • 373
Issue_Category table • 374
Issue_Property table • 376
Issue_Status table • 377
issues by number in messages • 79
- K**
- Key_Control table • 382
kmlrel object • 609
Knowledge Management
 keywords table • 383
 table for relating keywords to solutions • 383
 table for request solutions • 293
Knowledge_Keywords table • 383
Knowledge_Lrel_Table table • 383
kwrld object • 611
- L**
- labels, customizing • 67
layout
 statement
 creating data fields • 41
 including literal text • 41
statements

detailed description • 41
FOOTER • 57
HEADER • 58
HEADER2 • 59
overview • 41
PAGE FOOTER • 60
PAGE HEADER • 61
PRINT • 62

lists
 creating • 742
 master • 743
 restricted • 743

literal text in reports • 41
loc object • 611
Location table • 385
logged-in users • 66
logical tables • 724
lr object • 614
Lrel_Table table • 385

M

macro object • 616
macro_type object • 617
Main menu • 65
Majic
 files • 211
 statements
 ATTRIBUTES • 736
 FACTORY • 740
 MODIFY • 745
 MODIFY FACTORY • 746
 OBJECT • 734
 STANDARD_LISTS • 742
 types of • 733
 syntax • 733
Managed_Survey table • 386
Manufacturer table • 388
mapping statements • 730
message
 customization • 73
 notification • 27
mfrmod object • 618

mgs object • 619
Mgs_Status table • 390
mgsalg object • 620
mgsstat object • 621
MLIST parameter • 743
MODIFY
 FACTORY statement • 746
modifying
 reports • 37
 the schema • 723
multiline reports • 50

N

Note_Board table • 391
notification
 messages, customizing • 73
methods
 customizing • 25, 33
 script • 33
 variables • 27
Notification_Urgency table • 392
Notify_Log_Header table • 393
Notify_Object_Attr table • 394
notque object • 623
noturg object • 624
nr object • 624
nr_com object • 629
NR_Comment table • 395
nrf object • 630
ntfl object • 631
NX_NTF variables • 27

O

object
 schema • 210
objects
 activity log • 77
 bhvtpl • 496
 bool • 501
 changing • 733, 745
 chg • 504
 chg_tpl • 507
 chgalg • 508
 chgcat • 510
 chgstat • 509
 close_entry • 511
 cnote • 520
 cnt • 521

cr • 527
cr_prp • 529
cr_prptpl • 530
cr_tpl • 531
crs • 532
crsol • 533
crsq • 534
crt • 535
ctab • 536
ctimer • 537
ctp • 538
dcon • 539
defining • 733, 734
dlgsrvr • 541
dmn • 542
doc_rep • 543
evt • 563
evtdly • 565
evtdlytp • 566
ext_entity_map • 567
grc • 583
grp_loc • 584
grpmem • 585
hier • 586
imp • 587
intfc • 588
iss • 589
iss_prp • 593
iss_tpl • 594
iss_wf • 595
issalg • 596
isscat • 597
issstat • 601
kmlrel • 609
kwrd • 611
listing • 742
loc • 611
lr • 614
macro • 616
macro_type • 617
mfrmod • 618
mgs • 619
mgsalg • 620
mgsstat • 621
notque • 623
noturg • 624
nr • 624
nr_com • 629
nrf • 630
ntfl • 631
options • 635
org • 636
pcat • 639
pcat_grp • 640
pcat_loc • 641
pcat_workshift • 641
perscnt • 642
position • 643
pri • 644
prod • 645
prp • 645
prptpl • 646
quick_tpl_types • 647
rc • 648
request • 67
rptm • 650
rptmeth • 651
rrf • 652
rss • 653
sdsc • 656
seq • 658
sev • 660
site • 662
slatpl • 663
state • 665
survey • 666
svy_ans • 669
svy_qtpl • 670
svy_ques • 671
svy_tpl • 672
svystat • 673
svytrk • 674
tskstat • 675
tskty • 676
tspan • 677
typecnt • 678
tz • 679
urg • 680
vpt • 687
wf • 688
wftpl • 689
wrkshft • 691
Options Manager
 architecture • 210
 viewer • 210
options object • 635
Options table • 399
org object • 636

-
- OSI
 type table • 347
- output program • 56
- P**
- P1 statements • 730
- page
 footer layout statement • 60
 header layout statement • 61
- painter32 command • 216
- passing arguments into reports • 37
- pcat object • 639
- Pcat_Group table • 400
- pcat_grp object • 640
- pcat_loc object • 641
- Pcat_Loc table • 401
- pcat_workshift object • 641
- Pcat_Workshift table • 402
- pdm_task utility • 48
- perscnt object • 642
- Person_Contacting table • 402
- phone notification • 27
- physical tables • 724
- position object • 643
- pri object • 644
- print layout statement • 62
- Priority table • 403
- Prob_Category table • 404
- prod object • 645
- Product table • 406
- Property table • 406
- Property_Template table • 407
- prp object • 645
- prptpl object • 646
- Q**
- query
 on scoreboard • 65
- Queued_Notify table • 409
- Quick_Template_Types table • 410
- quick_tpl_types object • 647
- R**
- rc object • 648
- recording
 escalation levels table • 325
- REL_ATTR parameter • 741
- Remote_Ref table • 410
- report
- customization • 37
- formatting flags • 50
- functions • 52
- generation • 47
- multiline • 50
- output redirector • 48
- passing arguments into • 37
- program • 48
- template • 39
- Reporting Method table • 411
- Req_Property table • 412
- Req_Property_Template table • 413
- Request type table • 292
- requests
 activity types table • 225
 by logged-in user • 66
 by number in messages • 78
 log table • 224
 object • 67
 prefixes table • 420
 request areas table • 404
 sequence table • 420
 service types table • 424
 smart hooks table • 410
 status table • 315
 stored queries table • 316
 suffixes table • 420
 table for recording users' issues • 287
 timers table • 314
- Resource_Family table • 414
- Reverse_Boolean_Table table • 415
- RLIST parameter • 743
- Rootcause table • 416
- Rpt_Meth table • 416
- rptm object • 650
- rptmeth object • 651
- rrf object • 652
- rss object • 653
- S**
- Schema customization
 syntax • 723
- Schema.sch file • 723
- scoreboard customization • 65
- Screen Painter
 adding controls • 217
 denying access to a functionality by added controls • 217

restricting functionality by added controls • 217
using • 209

script for notification method • 33

sdsc object • 656

select clause • 56

seq object • 658

Sequence_Control table • 420

Service_Desc table • 424

Service_Level_Agreement table • 425

sev object • 660

Severity table • 427

site

- directory • 211
- mods

- directory • 211

majic directory • 211

object • 662

Site table • 427

SLA_Progress_Notification_Log table • 432

SLA_Template table • 433

slatpl object • 663

sort clause • 56

SORT_BY parameter • 743

Spell_Macro table • 434

SQL clauses

- ALIAS keyword • 56
- SELECT • 56
- SORT • 56
- WHERE • 56

SQL_Script table • 436

STANDARD_LISTS statement • 742

Starting

- Screen Painter • 216

state object • 665

stored queries • 65

survey object • 666

Survey table • 437

Survey_Answer table • 438

Survey_Answer_Template table • 439

Survey_Question table • 440

Survey_Question_Template table • 441

Survey_Stats table • 442

Survey_Template table • 443

Survey_Tracking table • 444

svy_ans object • 669

svy_qtpl object • 670

svy_ques object • 671

svy_tpl object • 672

svystat object • 673

svytrk object • 674

T

TABLE statements • 724

TABLE_INFO statements • 728

tables

- A_Comment • 243, 318
- Access_Levels • 219
- Access_Type • 220
- Act_Log • 224
- Act_Type • 225
- Act_Type_Assoc • 229
- Active_Boolean_Table • 230
- Active_Reverse_Boolean_Table • 231
- Animator • 234
- Attached_Events • 238
- Attachment • 241
- Attribute_Name • 244
- Audit_Log • 245
- Behavior_Table • 246
- Boolean_Table • 248
- Bop_Workshift • 248
- Call_Req • 287
- Call_Req_Type • 292
- Call_Solution • 293
- Change_Act_Log • 294
- Change_Category • 296
- Change_Status • 303
- Chg_Template • 304
- Computer_Extension • 305
- Controlled_Table • 313
- Cr_Call_Timers • 314
- Cr_Status • 315
- Cr_Stored_Queries • 316
- Cr_Template • 317
- Delegation_Server • 319
- Document_Repository • 320
- Domain • 321
- Domain_Constraint • 322
- Domain_Constraint_Type • 323
- Escalation_Control • 324
- Escalation_Entry_Log • 325
- Event_Delay • 340
- Event_Delay_Type • 341
- Events • 343
- Expense_Code_Opt • 345
- External_Entity_Map • 345

Form_Group • 346
Geo_Coord_Type • 347
Impact • 363
Interface • 364
ISS_Template • 365
Issue • 368
Issue_Act_Log • 373
Issue_Category • 374
Issue_Property • 376
Issue_Status • 377
Issue_Workflow_Task • 378
Key_Control • 382
Knowledge_Keywords • 383
Knowledge_Lrel_Table • 383
Location • 385
Lrel_Table • 385
Managed_Survey • 386
Manufacturer • 388
Mgs_Status • 390
Note_Board • 391
Notification_Urgency • 392
Notify_Log_Header • 393
Notify_Object_Attr • 394
NR_Comment • 395
Options • 399
OSI_type • 347
Pcat_Group • 400
Pcat_Loc • 401
Pcat_workshift • 402
Person>Contacting • 402
Priority • 403
Prob_Category • 404
Product • 406
Property • 406
Property_Template • 407
Queued_Notify • 409
Quick_Template_Types • 410
Remote_Ref • 410
Reporting_Method • 411
Req_Property • 412
Req_Property_Template • 413
Resource_Family • 414
Reverse_Boolean_Table • 415
Rootcause • 416
Rpt_Meth • 416
Sequence_Control • 420
Service_Desc • 424
Service_Level_Agreement • 425
Severity • 427
Site • 427
SLA_Progress_Notification_Log • 432
SLA_Template • 433
Spell_Macro • 434
Spell_Macro_Type • 435
SQL_Script_table • 436
Survey • 437
Survey_Answer • 438
Survey_Answer_Template • 439
Survey_Question • 440
Survey_Question_Template • 441
Survey_Stats • 442
Survey_Template • 443
Survey_Tracking • 444
Table_Name • 445
Task_Status • 446
Task_Type • 451
Timespan • 448
Timezone • 449
Transition_Points • 450
Type_Of_Contact • 452
Urgency • 453
User_Query • 454
Workflow_Task • 465
Workflow_Task_Template • 468
Tag Helper, function definition • 212
tags
 overview • 212
Task_Status_table • 446
Task_Type_table • 451
telephone notification • 27
text in reports • 41
Timespan_Table • 448
Timezone_table • 449
Transition_Points_table • 450
tskstat object • 675
tskty object • 676
tspan object • 677
Type_Of_Contact_table • 452
typecnt object • 678
tz object • 679

U

urg object • 680
Urgency_table • 453
User_Query_table • 454
user-defined code table • 230, 231
using Screen Painter • 209
utilities

for report generation • 48
pdm_task • 48

V

variable
expressions • 43, 49
in notification method • 27
passing into reports • 37
voice mail notification • 27
vpt object • 687

W

wf object • 688
wftpl object • 689
WHERE clause • 56
Workflow_Task table • 465
Workflow_Task_Template table • 468
workshifts for events table • 248
wrkshft object • 691