RightNow CRM™ 7.5

Integration Manual

July 2005

RIGHT NOW CRM

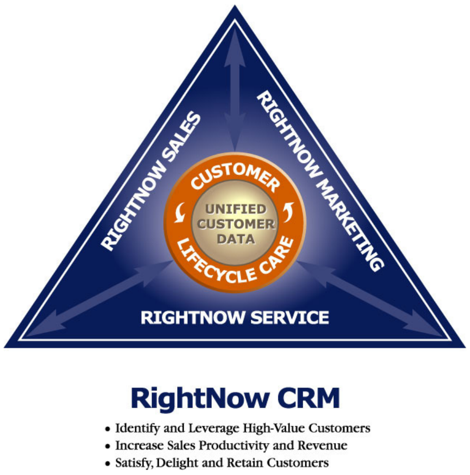
Web address: http://rightnow.com

Email address: info@rightnow.com

Contents

1 Introduction

Successful implementation of RightNow CRM is critical to the success of just about every organization. Customers, after all, are any organization's most important asset. RightNow CRM ensures this asset is fully leveraged by optimizing sales, marketing, and service interactions. With effective RightNow CRM, customer relationships can be appropriately managed to maximize revenue and lifespan while keeping operational costs low.


RIGHTNOW SALES
RIGHTNOW MARKETING
RIGHTNOW SERVICE
CUSTOMER LIFECYCLE CARE
UNIFIED CUSTOMER DATA
RightNow CRM
• Identify and Leverage High-Value Customers
• Increase Sales Productivity and Revenue
• Satisfy, Delight and Retain Customers

Our latest release, RightNow CRM, expands our on-demand CRM offerings focused on customer service to fully integrate CRM business processes across sales, marketing, and customer service departments. By doing so, RightNow CRM ensures that customer interactions with any department work to the advantage of all customers.

RightNow Marketing is an outbound email and campaign solution for marketing professionals looking to leverage vast amounts of information about customers and prospects gained from sources as diverse as online inquiries, service interactions, ad responses, and sales calls. This solution, with out-of-the-box integration with RightNow Service and RightNow Sales, empowers you to deliver individualized, targeted outbound email and campaigns to highly qualified customers and prospects. The result is a deeper customer relationship, more qualified responses, and greater customer loyalty—all resulting in increased sales.

RightNow Sales is an automated solution designed to maximize your sales performance while providing your customers the best possible experience. RightNow Sales includes all the functionality your sales professionals demand from a world-class sales automation solution to automate business processes for improved predictability and accountability throughout the entire organization. With out-of-the-box integration with RightNow Service and RightNow Marketing, RightNow Sales provides all departments access to the same customer data and brings your organization closer to its CRM goals of shared customer data across sales, service, and marketing departments

About this manual

This manual is targeted for RightNow administrators and experienced programmers with extensive knowledge of RightNow CRM and the external program used for integration. For integration that involves manipulation of data, you must be a non-hosted customer operating the latest version of RightNow CRM (7.5 or higher). Customers hosted by RightNow Technologies must contact their RightNow Technologies account manager for integration assistance.

The information contained in this manual includes the following:

Chapter 2, Integration Overview—Provides an overview of the three methods of integrating RightNow CRM with other CRM (customer relationship management) applications.

Chapter 3, XML Integration—Contains information about using Extensible Markup Language (XML) to integrate RightNow CRM with external systems. Included are descriptions of the XML tags, API functions, and methods of integration.

Chapter 4, Event Handlers—Contains procedures for using RightNow CRM's external event interface to define custom processes for managing your knowledge base records.

Chapter 5, Pass-Through Authentication—Contains procedures for passing login parameters from an external customer validation source to RightNow Service through a URL containing the data.

Appendix A, Pair Names—Contains the API pair names that describe the database field or other information passesd to and from the RightNow CRM API.

Appendix B, Source Codes—Contains the source codes that can be used when creating contacts, incidents, opportunities, and organizations.

Appendix C, Logical Bit Flags—Contains the logical bit controls for actions that occur within the API functions.

Appendix D, Database Schema Tables—Contains the schema tables of the RightNow database.

RightNow Technologies documentation

RightNow documentation includes several manuals and documents to help you install, administer, and use RightNow products, including RightNow CRM™ (RightNow Service™, RightNow Marketing™, and RightNow Sales™), RightNow Metrics™, and RightNow Locator™. Our documentation is written for RightNow users who have a working knowledge of their operating system and Web browsers and are familiar with standard conventions such as using menus and commands to open, save, and close files.

Note RightNow documentation is available in printed and/or PDF formats. Certain manuals and documents are available in PDF format only and are clearly noted in this documentation list. To download a document or manual in PDF format, or to order extra copies of printed documentation, contact your RightNow account manager.

RightNow CRM

The following RightNow CRM documentation contains system configuration information that is common to RightNow Service, RightNow Marketing, and RightNow Sales.

RightNow CRM System Specifications—Contains a description of the recommendations and requirements for RightNow CRM user workstations, along with Web server, database server, and mail server recommendations and requirements for non-hosted customers. Refer to answer ID 31 on our customer support site at www.rightnow.com.

RightNow CRM Release Notes—Contains a brief description of the new and expanded features in the RightNow Service, RightNow Marketing, and RightNow Sales modules. This document is available in PDF format only.

RightNow CRM Installation Guide—Contains procedures for installing RightNow CRM on UNIX and Windows platforms. This guide is available in PDF format only.

RightNow CRM System Configuration Manual—Contains system configuration information common to the RightNow Service, RightNow Marketing, and RightNow Sales modules. These configuration options include staff management, customizable menus, custom fields, communication, business rules, views, computer telephony integration, and multiple interfaces. Also included are descriptions of utilities and system configuration options.

RightNow CRM Analytics Manual—Contains procedures for generating standard and custom reports. Also included are descriptions of the RightNow Service, RightNow Marketing, and RightNow Sales standard reports.

RightNow CRM Integration Manual—Contains procedures for integrating the RightNow CRM knowledge base with external systems, including help desks, data mining, and data reporting systems. Contact your RightNow account manager for information on obtaining this manual. This guide is available in PDF format only.

RightNow Service HMS Guide—Contains upgrade instructions for customers hosted by RightNow Technologies. This guide is available in PDF format only.

RightNow CRM SmartConversion Guide—Contains procedures for upgrading from RightNow eService Center 5.5.7.1 to RightNow Service 7.0 along with procedures for upgrading from RightNow Service 6.0.3 to RightNow Service 7.0. This document also contains procedures for upgrading from RightNow Outbound 6.0.3 to RightNow Marketing 7.0. This guide is available in PDF format only.

RightNow CRM Entity Relationship Diagram—Provides a diagram of the relationship of the tables and columns included in RightNow CRM, which include the RightNow Service, RightNow Marketing, and RightNow Sales modules.

RightNow Service

The following documentation is available for RightNow Service.

RightNow Service New and Expanded Features—Contains a detailed description of the new and expanded features included in this version of RightNow Service. This document is available in PDF format only.

RightNow Service Administration Manual—Contains procedures for configuring and customizing RightNow Service. Configuration options include customizable menus, custom fields, Configuration Wizard, communication configuration, service quality, content library, views, Support Console customization, answer creation and maintenance, Answer Console, end-user interface configuration and customization, Offer Advisor, RightNow Live, reports, RightNow Wireless, and incident archiving.

RightNow Service Agent Manual—Contains information and procedures for agents on using RightNow Service as their customer service and support solution. The information includes adding and editing incidents, contacts, and organizations along with procedures for using the Offer Advisor to promote offers and products to customers, and procedures for using the computer telephony integration (CTI) system for answering incoming calls. Also included is a detailed description of the Support Console and end-user interface and procedures for using RightNow Live.

Click the Help tab on the end-user pages for information about searching through the end-user pages of RightNow Service.

For help within RightNow Service, click . For help on functions within a window or console, to open the help contents, or to open the help index. Also click this button to access the online documentation.

RightNow Marketing

The following documentation is available for RightNow Marketing.

RightNow Marketing New and Expanded Features—Contains a detailed description of the new and expanded features included in this version of RightNow Marketing. This document is available in PDF format only.

RightNow Marketing Administration Manual—Contains procedures for configuring and customizing RightNow Marketing. Configuration options include communication configuration, customizable menus, views, reports, and procedures for using the Contact Uploader to upload contact information into the RightNow database.

RightNow Marketing User Manual—Contains procedures for creating and managing marketing campaigns and distributing outbound email. Also included are detailed descriptions of each RightNow Marketing console.

For help within RightNow Marketing, click . For help on functions within a window or console, to open the help contents, or to open the help index. Also click this button to access the online documentation.

RightNow Sales

The following documentation is available for RightNow Sales.

RightNow Sales Administration Manual—Contains procedures for configuring and customizing RightNow Sales. Configuration options include customizable menus, custom fields, monetary configuration, sales process, quotes, views, and Sales Console customization.

RightNow Sales User Manual—Contains procedures for using RightNow Sales to automate and manage your sales processes. Also included is a detailed descriptions of the Sales Console.

For help within RightNow Sales, click . For help on functions within a window or console, to open the help contents, or to open the help index. Also click this button to access the online documentation.

RightNow Metrics

The following documentation for RightNow Metrics is available in PDF format only.

SmartStart Surveys—Contains valuable information about developing surveys using RightNow Metrics. It includes the steps necessary to plan your survey strategy and also provides helpful tips for creating successful surveys.

RightNow Metrics Release Notes—Contains a brief description of the new and expanded features of RightNow Metrics. For a more detailed description of these features, refer to *RightNow Metrics New and Expanded Features.*

RightNow Metrics New and Expanded Features—Contains descriptions of the new and expanded features of RightNow Metrics.

RightNow Metrics Installation Guide—Contains procedures for installing RightNow Metrics.

RightNow Metrics User Manual—Contains procedures for creating, launching, and maintaining RightNow Metrics surveys. Also included are procedures for integrating RightNow Metrics with RightNow eService Center, procedures for displaying survey results, and procedures for scheduling utilities. Appendices include the configuration settings, procedures for accessing the message bases, a list of predefined questions and surveys, and the database schema tables.

Click the Links drop-down menu on most administration pages to access online help.

RightNow Locator

The following documentation for RightNow Locator is available in PDF format only.

RightNow Locator Release Notes—Contains a brief description of the new and expanded features in this version of RightNow Locator.

RightNow Locator Administration Manual—Contains procedures for configuring and customizing RightNow Locator.

RightNow Locator User Manual—This manual contains procedures for configuring RightNow Locator to enable your customers to quickly find location information, and access maps and driving directions. Also included are procedures for configuring RightNow Locator so that your customers can chat with a RightNow Locator agent, or talk to a location directly through their PC.

RightNow Locator Advanced Customization through the Extended File Manager—This document contains descriptions of the files and directories available through the Expanded File Manager, which can be used to customize the RightNow Locator end-user pages. Contact your sales representative for information on obtaining this document.

Click the Links drop-down menu on most administration consoles to access the online documentation.

RightNow Locator SmartConversion Guide—Contains procedures for upgrading RightNow Locator 2.0 to RightNow Locator 3.0, including the necessary updates and the areas affected by the upgrade.

RightNow Locator XML Guide—This manual contains procedures for configuring RightNow Locator to enable your customers to quickly find location information, and access maps and driving directions. Also included are procedures for configuring RightNow Locator so that your customers can chat with a RightNow Locator agent, or talk to a location directly through their PC.

2 Integration Overview

RightNow CRM has all the tools you need to create a fully integrated customer service solution. There are three ways to integrate RightNow CRM with other applications:

- XML API
- Event handlers
- Pass-through authentication

This overview provides a brief description of each integration method to assist you in deciding which method best suits your integration needs. For detailed information about the types of integration, refer to each method's chapter in the manual.

You must be a non-hosted customer to access the API through XML or implement event handlers. If you are a hosted customer, you must contact your RightNow account manager to perform these functions. Both hosted and non-hosted customers may contact their account manager for assistance from Professional Services in planning and implementing an integration. To learn more about the services provided, visit our web site at:

```
http://rightnow.com
```

To follow the procedures in this manual, non-hosted customers must be using the latest version of RightNow CRM (7.5 or higher), available for download on our web site.

Caution The API functions should be used by experienced programmers only. Misuse of the API could result in damage to your RightNow CRM site or database. We recommend that you first test your integration on a non-production site. If you require assistance, contact your RightNow account manager.

XML API

XML integration allows you to interact directly with the API through the use of XML. Through XML integration, you can create, update, delete, get, and search on accounts, answers, contacts, hierarchical menus, incidents, meta-answers, opportunities, organizations, quotes, SLA instances, and task instances in your RightNow database. You can also run SQL queries on any table of your RightNow database to retrieve information. You can use HTTP POST or send an XML-formatted email to perform XML API functions. Posting XML allows real-time interaction with RightNow CRM.

Use XML integration when you want direct access to the RightNow database. XML integration can also be used when an external application has the ability to create and send XML-formatted emails or post XML directly to RightNow CRM. You should have experience with XML and familiarity with RightNow CRM functions before attempting to perform an XML integration. For more information, refer to Chapter 3, "XML Integration," on page 17.

Event handlers

An event handler is implemented when a specific event occurs within RightNow CRM. An event handler can either execute a script (external event) or email data (application bridge) to a specified email address when the event occurs. The following events are supported:

- An incident is created, updated, or deleted
- An answer is created, updated, or deleted
- A contact is created, updated, or deleted
- An organization is created, updated, or deleted
- An opportunity is created, updated, or deleted
- A business rule is matched

These events facilitate execution of a program or email transmission when information is modified within RightNow CRM. The external event program is passed the data related to the update. For example, this function could be used to update contact information in an external system every time contact information is updated within RightNow CRM.

Use event handlers when you want real-time synchronization with an external system or want to update data external to RightNow CRM. Using external events requires programming experience and familiarity with RightNow CRM functions. For more information, refer to Chapter 4, "Event Handlers," on page 81.

Pass-through authentication

You can integrate RightNow Service with an external customer validation source to allow your end-users to automatically log in to RightNow Service from an external web page by passing the login parameters in the URL of any appropriate end-user page (home.php, std_alp.php, std_adp.php). By using this integration method, you can allow contacts to have one login name and password for RightNow Service, as well as an external system.

Use this integration method when you want to use an external customer validation source to log in contacts to RightNow Service. This method requires programming experience and familiarity with RightNow Service functions. For more information, refer to Chapter 5, "Pass-Through Authentication," on page 89.

3

XML Integration

You can use XML (Extensible Markup Language) to access RightNow CRM's API and update the database. Through XML integration, you can perform many tasks normally accomplished through the RightNow user interfaces, such as creating, updating, deleting, retrieving and searching records in your RightNow database using either of the following methods:

- **Sending XML data using the POST method**—When posting data using this method, the XML is immediately passed to RightNow CRM and parsed by a PHP script. A record is then instantly created, updated, deleted, retrieved, or searched for in the RightNow database. For additional information, refer to "Using the POST method" on page 69.
- **Sending an XML-formatted email**—When sending an XML-formatted email, the utility *techmail* will identify an email as having XML through a trigger word or phrase in the subject line. The email will then be parsed by a PHP script to retrieve the data. For additional information, refer to "Sending an XML-formatted email" on page 70.

Caution The XML API functions should be used by experienced programmers only. Misuse of the API could result in damage to your RightNow CRM site or database. We recommend that you first test your integration on a non-production site. If you require assistance, please contact your RightNow account manager.

This chapter describes the XML tags used by the RightNow CRM API, provides descriptions of the basic API functions, and contains information on posting XML through a URL or sending an XML-formatted email.

XML tags

The data sent to RightNow CRM is identified by a series of XML tags defined in this chapter. The tags organize data so the it can be parsed by RightNow CRM and handled appropriately. There are four basic tags used when accessing the API through XML:

- <connector>
- <function>
- <parameter>
- <pair>

These tags are described in the following sections, along with descriptions of their attributes and types.

<connector> tag

The <connector> tag is the root element of the XML code. It contains all function tags. The <connector> tag can use the following attributes:

- **ret_type**—This attribute specifies either http or email as the type of return. For example:

```
<connector ret_type="http">
```

If the ret_type is set to email, an email containing the return value will be sent. If the ret_type is set to http, the XML return value will be sent to the http requester. If no ret_type is specified, http will be used by default.

Note Specifying `ret_type="http"` does not allow you to send an XML return to a specific http location or URL.

- **ret_email**—This attribute specifies the email address to send return values to if ret_type is email. For example:

```
<connector ret_type="email" ret_email="jdoe@isp.com">
```

When return values are sent to an email address or URL, they appear in the following format:

```
<?xml version="1.0" encoding="UTF-8"?>
<connector_ret>
     <function name="incident_create">
          <ret_val name="i_id">7345</ret_val>
     </function>
</connector_ret>
```

This example returns the i_id of an incident created through the API with the incident_create function. The automatically assigned i_id of the new incident, 7345, is specified by the <ret_val> tag.

<function> tag

The <function> tag contains each API call and contains the following attributes:

- **name**—This attribute specifies the name of the API function you want to call. For example:

  ```
  <function name="incident_create">
  ```

- **id**—This attribute specifies a string used to apply return values to. The string can be used later to have the return value replace a variable. For example:

  ```
  <function name="contact_create" id="contactid">
  ```

  For information about using variables in the ID attribute, refer to “Passing variable IDs” on page 74.

<parameter> tag

The following of <parameter> options can be specified using the name attribute. The datatype should also be specified using the type attribute. All of the following parameters are integer datatypes, except args (which is a pair) and lk_str (which is a string).

- **a_id**—This parameters defines the answer ID for answer API functions. The a_id will be created by the API and returned by the function. For example:

  ```
  <parameter name="a_id" type="integer">31</parameter>
  ```

- **acct_id**—This parameters defines the account ID for account API functions. The acct_id will be created by the API and returned by the function.
- **args**—This parameters indicates that pair data will follow the <parameter> tag.

  ```
  <parameter name="args" type="pair"><pair data></parameter>
  ```

- **c_id**—This parameters defines the contact ID for contact API functions. The c_id will be determined by the API and returned by the function.
- **campaign_id**—This parameters defines the campaign ID for contact API functions. The campaign_id will be determined by the API and returned by the function.
- **flags**—This parameters defines the hex value of a logical bit flag. Refer to Appendix C, “Logical Bit Flags,” on page 131 for available flags.

```
<parameter name="flags" type="integer">0x00002</parameter>
```

This example calls the CALL_EXTERNAL_EVENT (0x00002) flag.

Note You must use the hex value of logical bit flags when performing an XML integration. If you are sending more than one logical bit flag for a function, add the values of the flags together. For example, when using the incident_update function you can send a logical bit flag to execute an external event (CALL_EXTERNAL_EVENT—0x00002) and a send a staff notification (NOTIFY_STAFF—0x00008) by passing the hex value 0x00010.

- **id**—This parameters defines the hierarchical menu ID for the hierarchical menu API functions. The ID will be created by the API and returned by the function.
- **i_id**—This parameters defines the incident ID for incident API functions. The i_id will be created by the API and returned by the function.
- **lk_fld**—This parameters defines the cf_id of the menu_item field you want to look up with the lookup_id_for_name function**.**
- **lk_str**—This parameters defines the name of the string you want to look up with the lookup_id_for_name function**.**
- **lk_tbl**—This parameters defines the table ID number that is associated with the field you want to look up with the lookup_id_for_name function. Table numbers are specified in Table 8 on page 72.
- **max_rows**—This parameters defines the maximum number of records that should be returned by the search.
- **m_id**—This parameters defines the meta-answer ID for meta-answer API functions. The m_id will be created by the API and returned by the function.
- **op_id**—This parameters defines the opportunity ID for opportunity API functions. The op_id will be created by the API and returned by the function.
- **org_id**—This parameters defines the organization ID for organization API functions. The o_id will be created by the API and returned by the function.
- **quote_id**—This parameters defines the quote ID for quote API functions. The quote_id will be created by the API and returned by the function.
- **slai_id**—This parameters defines the SLA instance ID for the SLA instance API functions. The slai_id will be created by the API and returned by the function.
- **sql**—This parameters defines the the SQL statement for the SQL query API functions.
- **ti_id**—This parameters defines the task instance ID for task API functions. The ti_id will be created by the API and returned by the function.

- **view_id**—This parameters defines the ID number of the view used to return records when performing a search.

<pair> tag

The <pair> tag contains data used by the API function. It describes the database field and the value to add to the RightNow database. The <pair> tag can have the following attributes:

- **name**—This attribute defines the pair name that the enclosed data pertains to. Pair names for the API are described in Appendix A, “Pair Names,” on page 103. For example:

  ```
  <pair name="title" type="string">Title</pair>
  ```

- **type**—This attribute determines whether the pair is a pair, integer, date, or string. For example:

  ```
  <pair name="source" type="integer">3</pair>
  ```

Table 1 describes the four pair types.

Table 1: Pair Type Description

Type	**Description**
integer	A positive or negative 4-byte integer.
string	A string of characters that cannot contain any NULLs.
time	A field that is the same type as the UNIX date_t, generally a long integer that is the number of seconds since the UNIX Epoch date (00:00:00 UTC January 1, 1970).
pair	This is not a specific type, but a pair that contains additional pair data.

Using special characters

When passing data through XML, there are certain characters that cannot be used because they are misinterpreted by the XML language as it is parsed in RightNow CRM. These special characters should always be encoded when used as a parameter value in your XML code. Table 2 shows the special characters and their required format.

Table 2: Special Characters

Character	**Format in XML**
&	&amp;

Table 2: Special Characters

Character	Format in XML
"	&quot;
’	&apos; or &#039;
<	&lt;
>	&gt;

XML API functions

When using XML to integrate RightNow CRM with an external system, you can use several API functions to perform actions upon accounts, answers, contacts, hierarchical menus, incidents, meta-answers, opportunities, organizations, quotes, SLA instances, and task instances. An XML API function is also available for searching in RightNow CRM.

Table 3 lists the available XML API functions along with a description of the function and its required parameter.

Table 3: XML API Functions

Function	Description	Required Parameters
acct_create	An account API function used to add an account to the database. Refer to page 28.	• Array of pair data
acct_destroy	An account API function used to delete an account from the database. Refer to page 29.	• Valid acct_id, seq, and group_id • Array of pair data
acct_move	An account API function used to move an account in the database. Refer to page 30.	• Valid tbl, sub_tbl, id, leaf, oldseq, newseq, oldparent, and newparent
acct_update	An account API function used to update an existing account in the database. Refer to page 31.	• Valid acct_id • Array of pair data
ans_create	An answer API function used to add an answer to the database. Refer to page 32.	• Array of pair data
ans_destroy	An answer API function used to delete an answer from the database. Refer to page 33.	• Valid a_id
ans_get	An answer API function used to retrieve an answer from the database. Refer to page 33.	• Valid a_id
ans_update	An answer API function used to update an existing answer in the database. Refer to page 34.	• Valid a_id • Array of pair data

Table 3: XML API Functions (Continued)

Function	Description	Required Parameters
campaign_execute	A campaign API function used to execute a marketing campaign. Refer to page 35. **Note:** This function should only be used if the Marketing module is enabled.	• Valid c_id, campaign_id, and entry_point
contact_create	A contact API function used to add a contact to the database. Refer to page 35.	• Array of pair data
contact_destroy	A contact API function used to delete a contact from the database. Refer to page 36.	• Valid c_id
contact_get	A contact API function used to retrieve a contact from the database. Refer to page 37.	• Valid c_id
contact_update	A contact API function used to update an existing contact in the database. Refer to page 37.	• Valid c_id • Array of pair data
mail_to_contact	A contact API function used with contact_create or contact_update for creating or updating a contact and sending them an email in the same XML post. Refer to page 38. **Note:** This function should only be used if the Marketing module is enabled.	• Valid c_id,mailing_id, and campaign_id
hiermenu_create	A hierarchical menu API function used to create hierarchical menu items in the database. Refer to page 39.	• Valid tbl, seq, name, lvl (0-5), parent_id, label, description, and visibility • Array of pair data
hiermenu_destroy	A hierarchical menu API function used to destroy hierarchical menu items in the database. Refer to page 40.	• Valid id, tbl, parent_id, lvl and seq • Array of pair data

Table 3: XML API Functions (Continued)

Function	Description	Required Parameters
hiermenu_move	A hierarchical menu API function used to move hierarchical menu items in the database. Refer to page 41.	• Valid tbl, id, oldseq, newseq, oldparent, newparent, oldlvl, newlvl, and nchildren
hiermenu_update	A hierarchical menu API function used to update an existing menu in the database. Refer to page 41.	• Valid id and tbl • Array of pair data
incident_create	An incident API function used to add an incident to the database. Refer to page 42.	• Array of pair data
incident_destroy	An incident API function used to delete an incident from the database. Refer to page 44.	• Valid i_id
incident_get	An incident API function used to retrieve a specific incident from the database. Refer to page 45.	• Valid i_id
incident_update	An incident API function used to update an existing incident in the database. Refer to page 45.	• Valid i_id • Array of pair data
lookup_id_for_name	A function used to look up the code number of a field in RightNow CRM. Refer to page 77.	• Valid lk_tbl, lk_str, and lk_fld
meta_ans_create	A meta-answer API function used to add a meta-answer to the database. Refer to page 46.	• Array of pair data
meta_ans_destroy	A meta-answer API function used to delete a meta-answer from the database. Refer to page 48.	• Valid m_id
meta_ans_update	A meta-answer API function used to update an existing meta-answer in the database. Refer to page 49.	• Valid m_id • Array of pair data

Table 3: XML API Functions (Continued)

Function	Description	Required Parameters
org_create	An organization API function used to add an organization to the database. Refer to page 54.	• Array of pair data
org_destroy	An organization API function used to delete an organization from the database. Refer to page 55.	• Valid org_id
org_get	An organization API function used to retrieve an organization from the database. Refer to page 56.	• Valid org_id
org_update	An organization API function used to update an existing organization in the database. Refer to page 56.	• Valid org_id • Array of pair data
sa_opp_create	An opportunity API function used to add an opportunity to the database. Refer to page 50.	• Array of pair data
sa_opp_destroy	An opportunity API function used to delete an opportunity from the database. Refer to page 51.	• Valid op_id
sa_opp_get	An opportunity API function used to retrieve an opportunity from the database. Refer to page 52.	• Valid op_id
sa_opp_update	An opportunity API function used to update an existing opportunity in the database. Refer to page 52.	• Valid op_id • Array of pair data
sa_quote_destroy	A quote API function used to delete a quote from the database. Refer to page 57.	• Valid quote_id
sa_quote_get	A quote API function used to retrieve a quote from the database. Refer to page 57.	• Valid quote_id

Table 3: XML API Functions (Continued)

Function	Description	Required Parameters
sa_quote_update	A quote API function used to update an existing quote in the database. Refer to page 58.	• Valid quote_id • Array of pair data
sa_task_ins_create	A task API function used to add a task instance to the database. Refer to page 66.	• Array of pair data
sa_task_ins_destroy	A task API function used to delete a task instance from the database. Refer to page 67.	• Valid ti_id
sa_task_ins_get	A task API function used to retrieve a task instance from the database. Refer to page 67.	• Valid ti_id
sa_task_ins_update	A task API function used to update an existing task instance in the database. Refer to page 68.	• Valid ti_id • Array of pair data
search	A search API function used to search for any records in the database using an existing view. Refer to page 58.	• Valid view_id
slai_create	An SLA API function used to create an SLA instance in the database. Refer to page 63.	• Array of pair data
slai_terminate	An SLA API function used to terminate an SLA instance in the database. Refer to page 64.	• Valid slai_id
sql_get_int	An SQL query API function used to retrieve an integer value from the database. Refer to page 64.	• SQL parameter and statement
sql_get_str	An SQL query API function used to retrieve a string from the database. Refer to page 65.	• SQL parameter and statement

Table 3: XML API Functions (Continued)

Function	Description	Required Parameters
sql_get_time	An SQL query API function used to retrieve a datetime value from the database. Refer to page 66.	• SQL parameter and statement

Important If an invalid parameter ID is used with the XML API _get functions, a blank return value will be returned.

Note The XML API _get functions do not return data fields with NULL values.

Account API

The account API functions (acct_create, acct_destroy, acct_move, and acct_update) allow you to create, delete, move, or update information within the *accounts* table. You can act on all standard database fields of the *accounts* table, as well as some specialized information, such as staff account custom fields.

To access staff accounts within RightNow CRM, click >Common Configuration>Staff Management>Staff Accounts.

acct_create

The acct_create function is used to add a staff account to the RightNow database. The function has one component: an array of pair data.

Important The API will automatically generate an acct_id for the account that is consistent with existing accounts in the database.

The account will be populated with data specified in the pair list. A brief description of all *accounts* table fields and their corresponding pair names can be found in Appendix A, “Pair Names,” on page 103.

Example:

```
<connector>
     <function name="acct_create">
          <parameter name="args" type="pair">
               <pair name="def_currency" type="integer">1</pair>
               <pair name="seq" type="integer">3</pair>
               <pair name="profile_id" type="integer">2</pair>
               <pair name="country_id" type="integer">1</pair>
```

```
                <pair name="notif_pending" type="integer">0</pair>
                <pair name="group_id" type="integer">1</pair>
                <pair name="login" type="string">cjones</pair>
                <pair name="first_name" type="string">Chad</pair>
                <pair name="display_name" type="string">Chad Jones</pair>
            </parameter>
        </function>
</connector>
```

This example creates an account in the *accounts* table and the account ID is automatically returned by the function.

Note If email_notif is set to 1, a valid email_address is required.

Important If Computer Telephony Integration (CTI) is enabled, the acd_group field is required.

acct_destroy

The acct_destroy function is used to delete an existing account in the RightNow database. This function requires the following pairs to be set: a valid acct_id, seq, and group_id. A valid acct_id of an existing account must be supplied. If the acct_id is not set, or no staff account exists with the supplied ID number, the function will abort with an error message. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

Example:

```
<connector>
    <function name="acct_destroy">
        <parameter name="args" type="pair">
            <pair name="acct_id" type="integer">3</pair>
            <pair name="seq" type="integer">3</pair>
            <pair name="group_id" type="integer">1</pair>
        </parameter>
    </function>
</connector>
```

This example deletes account ID 3 from the database.

acct_move

The acct_move function is used to move an account from one group to another within the *accounts* table. The parameters of this function include tbl, sub_tbl, id, leaf, oldseq, newseq, oldparent, and newparent. The type attribute for each of these parameters is integer. A valid account ID of an existing account must be supplied. If the account ID is not set, or no account exists with the supplied acct_id, the function will abort with an error message.

Example:

```
<connector>
     <function name="acct_move">
          <parameter name="tbl" type="integer">24</parameter>
          <parameter name="sub_tbl" type="integer">0</parameter>
          <parameter name="id" type="integer">6</parameter>
          <parameter name="leaf" type="integer">1</parameter>
          <parameter name="oldseq" type="integer">5</parameter>
          <parameter name="newseq" type="integer">2</parameter>
          <parameter name="oldparent" type="integer">4</parameter>
          <parameter name="newparent" type="integer">2</parameter>
     </function>
</connector>
```

This example moves account ID 6 within the accounts table (24) from the fifth position in the hierarchy to the second position in the hierarchy and assigns it a new group (2).

Important If an account is referenced by a rule, it cannot be moved.

acct_update

The acct_update function is used to update the information associated with an existing staff account in the RightNow database. The function has one component: an array of pair data that includes a valid acct_id. If the function executes without error, a 1 will be returned. If the acct_id is not set, or no account exists with the supplied account ID, a zero will be returned. If an error occurs, a -1 will be returned.

Note If the Sales module is enabled and you want to change an account's territory, you must pass the old territory with the old_terr pair and the new territory ID with the terr_id pair. You will also need to specify whether to update the account's opportunities by setting the upd_opt pair to one of the following values.

1 - Update all opportunities

2 - Update only active opportunities

3 - Update no opportunities

The API will set any fields supplied in the pair list. Any staff account fields missing from the pair list will not be altered in the database.

Example:

```
<connector>
     <function name="acct_update">
          <parameter name="args" type="pair">
               <pair name="acct_id" type="integer">4</pair>
               <pair name="last_name" type="string">Jones</pair>
               <pair name="display_name" type="string">Chad Jones</pair>
          </parameter>
     </function>
</connector>
```

This example updates the account with an acct_id of 4 to have a last name of "Jones" and a display name of "Chad Jones."

Answer API

The answer API functions (ans_create, ans_destroy, ans_get, and ans_update) allow you to create, delete, retrieve, or update information from the *answers* table. You can act on all standard database fields of the *answers* table, as well as some specialized information, such as Answer custom fields.

On the administration side of RightNow CRM, answers reside on the Answer Console and may be designated for viewing by end-users. On the end-user pages, answers that have been designated for viewing by end-users appear on the Find Answers and Answer pages. Answer also refers to any knowledge base information that provides solutions to common customer support questions.

ans_create

The ans_create function is used to add an answer to the RightNow database. The function can have two components: an array of pair data and logical flag information.

Important The API will automatically generate an a_id for the answer that is consistent with existing answers in the database.

The answer will be populated with data specified in the pair list. A brief description of all *answers* table fields and their corresponding pair names can be found in Appendix A, "Pair Names," on page 103.

There are a number of components that must be included to create an answer. You must include a solution and description for the answer, the language ID, and an m_id (meta-answer ID).

◇Tip You can pass an existing meta-answer ID in the ans_create function using the m_id pair. You can also create a meta-answer and answer at the same time by passing the meta-answer ID from the meta-answer create function to the ans_create function.

Example:

```
<connector>
     <function name="ans_create">
          <parameter name="args" type="pair">
               <pair name="access_mask" type="string">1</pair>
               <pair name="assgn_acct_id" type="integer">6</pair>
               <pair name="description" type="string">How do I send a
               picture with my new camera phone?</pair>
               <pair name="m_id" type="integer">25</pair>
               <pair name="lang_id" type="integer">1</pair>
               <pair name="solution" type="string">Dear Valued Customer:
               Simply use the picture taking ability of your camera phone
               to take a photo. You can save the pictures on your phone
               up to the limit of the storage space on your camera and
               then send them to any World Mobile customer, upload them to
               your online photo album at World Mobile, or send them to an
               email address. To send to another World Mobile member, dial
               their number and select the Send Picture option on
```

```
                 your phone.</pair>
                 <pair name="status_id" type="integer">4</pair>
                 <pair name="summary" type="string">How do I send a
                 picture with my new camera phone?</pair>
            </parameter>
       </function>
</connector>
```

This example creates an answer associated with meta-answer ID 25 and sets the description, solution, and status fields for the answer. The function will return the a_id number.

ans_destroy

The ans_destroy function is used to delete an existing answer in the RightNow database. The function can have two components: the a_id of the answer and logical flag information. A valid a_id of an existing answer must be supplied. If the function executes without error, a 1 will be returned. If the a_id is not set, or no answer exists with the supplied ID number, the function will abort with an error message. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

Note Deleting the last answer under a meta-answer will also delete the meta-answer.

Example:

```
<connector>
     <function name="ans_destroy">
          <parameter name="a_id" type="integer">33</parameter>
          <parameter name="flags" type="integer">0x00002</parameter>
     </function>
</connector>
```

This example deletes answer ID number 33 and executes an external event.

ans_get

The ans_get function is used to retrieve the contents of an answer from the *answers* table. The single component of this function is the a_id number. A valid answer ID number of an existing answer must be supplied. If a valid a_id is not supplied, a blank return value will be returned

Example:

```
<connector>
     <function name="ans_get">
          <parameter name="a_id" type="integer">33</parameter>
     </function>
</connector>
```

This example retrieves the details for the answer with ID number 33 from the database.

Note The XML API ans_get function does not return data fields with NULL values.

ans_update

The ans_update function is used to update the information associated with an existing answer in the RightNow database. The function can have three components: the a_id of the answer, an array of pair data, and logical flag information. A valid answer ID of an existing answer must be supplied in the parameter tag. If the function executes without error, a 1 will be returned. If the a_id is not set, or no answer exists with the supplied answer ID, the function will abort with an error. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

The API will set any fields supplied in the pair list, including custom fields and meta-answer association. Any answer fields missing from the pair list will not be altered in the database.

Example:

```
<connector>
     <function name="ans_update">
          <parameter name="a_id" type="integer">33</parameter>
          <parameter name="args" type="pair">
               <pair name="access_mask" type="string">1</pair>
               <pair name="status_id" type="integer">4</pair>
          </parameter>
     </function>
</connector>
```

This example updates the answer with an a_id of 33 to have a status of Public (code 4) and access of Everyone (code 1).

Campaign API

The campaign API function campaign_execute allows you to execute a campaign in RightNow Marketing. Campaigns are multi-step email marketing processes based on business logic.

campaign_execute

The campaign_execute function is available when the marketing module is enabled. The function is used to send a specified contact through a campaign starting at the campaign entry point. The parameters of this function include c_id, campaign_id, and entry_point. The entry_point parameter corresponds to the Shortcut ID field of an entry point in a campaign. A campaign may have multiple entry points; however, each entry point's shortcut ID for will be unique.

Example:

```
<connector>
  <function name="campaign_execute">
    <parameter name="c_id" type="integer">2</parameter>
    <parameter name="campaign_id" type="integer">4</parameter>
    <parameter name="entry_point" type="string">ep1</parameter>
  </function>
</connector>
```

This example sends the contact with an ID of 2 through the campaign with an ID of 4, starting at the entry point named ep1.

Contact API

The contact API functions (contact_create, contact_destroy, contact_get, and contact_update) allow you to create, delete, retrieve, or update information from the *contacts* table. You can act on all standard database fields of the *contacts* table, as well as some specialized information, such as contact custom fields.

A contact is a customer who has a record in your RightNow CRM knowledge base. Contacts have a customer account which allows them to log in and submit incidents, subscribe to answer notifications, and view their recently submitted incidents.

Note For sales contacts, thread entries can be added or updated for contact notes, email, phone and notes field. These fields appear on the Notes tab in the contact record. For more information refer to "Thread entry types" on page 72.

contact_create

The contact_create function is used to add a contact to the RightNow database. The function can have two components: an array of pair data and logical flag information.

Important The API will automatically generate a c_id for the contact that is consistent with existing contacts in the database.

The contact will be populated with data specified in the pair list. A brief description of all *contacts* table fields and their corresponding pair names can be found in Appendix A, "Pair Names," on page 103. A brief description of all contact source codes can be found in Appendix B, "Source Codes," on page 127.

You must include a first name, last name, and CRM states for the contact in your XML. The email address of the contact is not required under certain configurations, such as call center applications. In this case, enabling CT_EMAIL_ENABLED would require the email address when using the contact_create function.

Example:

```
<connector>
     <function name="contact_create">
          <parameter name="args" type="pair">
               <pair name="first_name" type="string">Joe</pair>
               <pair name="last_name" type="string">Smith</pair>
               <pair name="css_state" type="integer">1</pair>
               <pair name="ma_state" type="integer">0</pair>
               <pair name="sa_state" type="integer">0</pair>
          </parameter>
          <parameter name="flags" type="integer">0x00002</parameter>
     </function>
</connector>
```

This example creates a contact, Joe Smith, who is associated with a service state, but not with a Marketing or Sales state. An external event is executed if one is specified in the EE_CONTACT_INSERT_HANDLER configuration setting. This function will return the contact ID from the database.

contact_destroy

The contact_destroy function is used to delete an existing contact in the RightNow database. The function can have two components: the c_id of the contact and logical flag information. A valid c_id of an existing contact must be supplied. If the function executes without error, a 1 will be returned. If the c_id is not set, or no contact exists with the supplied ID number, the function will abort with an error message. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

Caution Deleting a contact also results in the deletion of all incidents and opportunities associated with the contact.

Example:

```
<connector>
```

```
     <function name="contact_destroy">
          <parameter name="c_id" type="integer">8</parameter>
          <parameter name="flags" type="integer">0x00002</parameter>
     </function>
</connector>
```

This example deletes the contact with contact ID number 8 from the database and executes an external event.

contact_get

The contact_get function is used to retrieve a record from the *contacts* table. The single component of this function is the c_id. A valid ID number of an existing contact must be supplied. If no valid c_id is supplied, a blank value will be returned.

Example:

```
<connector>
     <function name="contact_get">
          <parameter name="c_id" type="integer">9</parameter>
     </function>
</connector>
```

This example retrieves the contact details with ID number 9 from the database.

Note The XML API contact_get function does not return data fields with NULL values.

contact_update

The contact_update function is used to update the information associated with an existing contact in the RightNow database. The function can have three components: the c_id, an array of pair data, and logical flag information. A valid contact ID of an existing contact must be supplied in the parameter tag. If the function executes without error, a 1 will be returned. If the c_id is not set, or no contact exists with the supplied contact ID, the function will abort with an error. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

The API will set any fields supplied in the pair list, including custom fields. Any contact fields missing from the pair list will not be altered in the database.

Example:

```
<connector>
     <function name="contact_update">
          <parameter name="c_id" type="integer">9</parameter>
          <parameter name="args" type="pair">
```

```
                <pair name="password" type="string">newpassword</pair>
            </parameter>
        </function>
</connector>
```

This example updates the contact with ID number 9. The contact's password is changed to "newpassword."

mail_to_contact

The mail_to_contact function is available when the Marketing module is enabled. This function can be used to send an event email to a contact. The parameters of this function include c_id, mailing_id, campaign_id, node_id, filter_id, and scheduled. The c_id, mailing_id, and campaign_id parameters are required. The scheduled parameter allows you to schedule an event email to be sent at a later date and time. The filter_id parameter allows you to specify a filter to prevent specified contacts from receiving event email messages. The filter_id corresponds to the *ma_events.filter_view_id* column in the RightNow database.

Example:

```
<connector>
  <function name="mail_to_contact">
    <parameter name="c_id" type="integer">84</parameter>
    <parameter name="mailing_id" type="integer">6</parameter>
    <parameter name="campaign_id" type="integer">5</parameter>
    <parameter name="filter_id" type="integer">133</parameter>
    <parameter name="scheduled"  type="time">1120478400</parameter>
  </function>
</connector>
```

This example would send the event email with the ID of 12 to the contact with the ID of 84, unless the contact was restricted by the filter with the ID of 133. The email will be sent the first time the mailer daemon runs after 12:00 PM on July 4, 2005 (UNIX timestamp of 1120478400).

Hierarchical menu API

The hierarchical menu API functions (hiermenu_create, hiermenu_destroy, hiermenu_move, and hiermenu_update) allow you to create, delete and alter hierarchical menus in RightNow CRM. For example, in RightNow Service you can act on products, categories, or dispositions and in RightNow Marketing you can act on outbound email categories and tracked link categories. You can act on all standard database fields of the *hier_menus* table.

Table 4 contains the hierarchical menus and corresponding table codes for use with the hierarchical menu API:

Table 4: Table Codes for Use in the Hierachical Menu API

Hierarchical Menu	Table Code
Service products	13
Service categories	14
Service dispositions	37
Marketing tracked link categories	56
Marketing outbound email categories	61

hiermenu_create

The hiermenu_create function is used to add a hierarchical menu item to the RightNow database. This function has one component: a pair array including valid tbl, seq, lvl (0-5), parent_id, label, desc, and vis pairs.

Important The API will automatically generate an ID for the hierarchical menu item that is consistent with existing objects in the database.

The hierarchical menu will be populated with data specified in the pair list. A brief description of all *hier_menus* table fields and their corresponding pair names can be found in Appendix A, “Pair Names,” on page 103.

Note The pairs listed in the following example are required.

Example:

```
<connector>
     <function name="hiermenu_create">
          <parameter name="args" type="pair">
               <pair name="lvl" type="integer">0</pair>
               <pair name="parent_id" type="integer"></pair>
               <pair name="seq" type="integer">1</pair>
               <pair name="tbl" type="integer">13</pair>
               <pair name="label" type="string">Wireless Plans</pair>
               <pair name="desc" type="string">This product folder lists
```

```
                  the different wireless plans available within the
                  organization.</pair>
                  <pair name="vis" type="pair">
                       <pair name="vis_item0" type="pair">
                            <pair name="admin" type="integer">1</pair>
                            <pair name="enduser" type="integer">1</pair>
                            <pair name="intf_id" type="integer">1</pair>
                       </pair>
                       <pair name="vis_item1" type="pair">
                            <pair name="admin" type="integer">0</pair>
                            <pair name="enduser" type="integer">0</pair>
                            <pair name="intf_id" type="integer">2</pair>
                       </pair>
                  </pair>
             </parameter>
        </function>
   </connector>
```

This example creates a top-level product, Wireless Plans, populates the description of the product, allows administration and end-user visibility on the interface with the ID of 1, and prohibits visiblity on the interface with the ID of 2.

hiermenu_destroy

The hiermenu_destroy function is used to delete an existing object (for example, a product or category) from a hierarchical menu. The function has one component: an array of pair data that includes a valid id, seq, tbl, parent_id, and lvl. All of these pairs are required. If the function executes without error, a 1 will be returned. If a pair is not set, or no hierarchical menu item exists with the supplied pair values, the function will abort with an error message. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

Example:

```
<connector>
     <function name="hiermenu_destroy">
          <parameter name="args" type="pair">
               <pair name="id" type="integer">7</pair>
               <pair name="seq" type="integer">7</pair>
               <pair name="tbl" type="integer">13</pair>
               <pair name="parent_id" type="integer"></pair>
               <pair name="lvl" type="integer">0</pair>
          </parameter>
     </function>
```

```
</connector>
```

This example deletes the product with the ID of 7 from the database.

hiermenu_move

The hiermenu_move function is used to move an object in a hierarchical menu. For example, you could move a top-level product to be a lower-level product under another top-level product. This function has nine parameters: tbl, id, oldseq, newseq, oldparent, newparent, oldlvl, newlvl, and nchildren. A valid hierarchical menu item ID of an existing object must be supplied. If the function executes without error, a 1 will be returned. If the ID is not set, or no menu exists with the supplied ID, the function will abort with an error message.

Example:

```
<connector>
     <function name="hiermenu_move">
          <parameter name="tbl" type="integer">13</parameter>
          <parameter name="id" type="integer">7</parameter>
          <parameter name="oldseq" type="integer">7</parameter>
          <parameter name="newseq" type="integer">7</parameter>
          <parameter name="oldparent" type="integer"></parameter>
          <parameter name="newparent" type="integer">6</parameter>
          <parameter name="oldlvl" type="integer">0</parameter>
          <parameter name="newlvl" type="integer">2</parameter>
          <parameter name="nchildren" type="integer">2</parameter>
     </function>
</connector>
```

This example moves the menu item with the ID of 7 to the third level (newlvl) in the hierarchical menu under a new hierarchical menu item menu (newparent), along with its associated lower-level hierarchical menu items.

Note The parameter *nchildren* describes the number of lower-level objects contained in the moved hierarchical menu item.

hiermenu_update

The hiermenu_update function is used to update the information associated with an existing hierarchical menu item in the RightNow database. The function has one component: an array of pair data that includes an existing ID and table. If the function executes without error, a 1 will be returned. If the ID is not set, or no account exists with the supplied ID, a zero will be returned. If an error occurs, a -1 will be returned.

The API will set any fields supplied in the pair list. Any hierarchical menu fields missing from the pair list will not be altered in the database.

Example:

```
<connector>
     <function name="hiermenu_update">
          <parameter name="args" type="pair">
               <pair name="id" type="integer">7</pair>
               <pair name="tbl" type="integer">13</pair>
               <pair name="name" type="string">Updated Product</pair>
          </parameter>
     </function>
</connector>
```

This example updates the name the product with the ID of 7 to be Updated Product.

Incident API

The incident API functions (incident_create, incident_destroy, incident_get, and incident_update) allow you to create, delete, retrieve, or update information from the *incidents* table. You can act on all standard database fields of the *incidents* table, as well as some specialized information, such as incident custom fields and incident threads.

An incident is a question or request for help from an end-user through any of the channels into RightNow Service, such as Ask a Question, email, Live chat, site or answer feedback, or the API.

incident_create

The incident_create function is used to add an incident to the RightNow database. The function has two components: an array of pair data and logical flag information.

Important The API will automatically generate an i_id for the incident that is consistent with existing incidents in the database.

The incident will be populated with data specified in the pair list. A description of all *incidents* table fields and their corresponding pair names can be found in Appendix A, "Pair Names," on page 103. When creating an incident, the source of the incident is a required element. For a listing of incident sources, refer to Appendix B, "Source Codes," on page 127.

Note Thread entries in incidents use a unique pair structure. For more information, refer to "Thread entry types" on page 72.

In addition to specifying the incident source, you must also include the incident contact ID number (c_id), title, and a thread. If you are using organizations, you should pass the organization ID number (org_id).

You may also need to specify products and categories if enabled in your configuration. You can choose to send an incident assignment notification, or send a receipt message to a contact by using a logical bit flag. Refer to Appendix C, “Logical Bit Flags,” on page 131.

Example:

```
<connector>
     <function name="incident_create">
          <parameter name="args" type="pair">
               <pair name="assgn_acct_id" type="integer">3</pair>
               <pair name="c_id" type="integer">7</pair>
               <pair name="cat_lvl1" type="integer">26</pair>
               <pair name="cat_lvl2" type="integer">27</pair>
               <pair name="interface_id" type="integer">1</pair>
               <pair name="lang_id" type="integer">1</pair>
               <pair name="org_id" type="integer">4</pair>
               <pair name="prod_lvl1" type="integer">2</pair>
               <pair name="prod_lvl2" type="integer">13</pair>
               <pair name="queue_id" type="integer">3</pair>
               <pair name="source" type="integer">3</pair>
               <pair name="status_id" type="integer">1</pair>
               <pair name="thread" type="pair">
                    <pair name="thread_entry" type="pair">
                         <pair name="entry_type" type="integer">3</pair>
                         <pair name="note" type="string">I want to send a
                         picture I have taken with my camera phone through
                         email. How can I do this?
                         </pair>
                    </pair>
               </pair>
               <pair name="subject" type="string">How do I send a picture
               with my new camera phone?</pair>
          </parameter>
          <parameter name="flags" type="integer">0x00002</parameter>
     </function>
</connector>
```

In this example, an incident is created for the contact with an ID of 7. The incident has a source of email (source code 3), and is unresolved (status code 1). The first-level product is set to code 2 and the second-level product is set to code 13. The first-level category is set to code 26 and the second-level category is set to code 27. In addition, a contact thread is created in the incident. The function returns the i_id number for the incident.

Important Because the 0x00002 logical bit flag parameter has been included, an external event will be executed if one is specified in the EE_INC_INSERT_HANDLER configuration setting.

Example:

```
<connector>
     <function name="incident_destroy">
          <parameter name="i_id" type="integer">7</parameter>
          <parameter name="flags" type="integer">0x00002</parameter>
     </function>
</connector>
```

This example deletes the incident with the ID of 7 from the database and executes an external event.

incident_destroy

The incident_destroy function is used to delete an existing incident. The function can have two components: the i_id of the incident and logical flag information. A valid ID number of an existing incident must be supplied. If the function executes without error, a 1 will be returned. If the i_id is not set, or no incident exists with the supplied ID number, the function will abort with an error message. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

Example:

```
<connector>
     <function name="incident_destroy">
          <parameter name="i_id">7</parameter>
          <parameter name="flags">0x00002</parameter>
     </function>
</connector>
```

This example deletes the incident with the ID of 7 from the database and executes an external event.

incident_get

The incident_get function is used to retrieve the contents of the *incidents* table. The required components of this function are the i_id of the incident and the logical bit flag for GET_THREADS (0x00200) if you want to include the incident threads in the output. A valid i_id of an existing incident must be supplied. If no valid i_id is supplied, a blank value will be returned.

Example:

```
<connector>
     <function name="incident_get">
          <parameter name="i_id" type="integer">1539</parameter>
          <parameter name="flags" type="integer">0x00200</parameter>
     </function>
</connector>
```

This example retrieves the incident details from the incident with the ID of 1539 from the database. All incident API pairs are returned using this function, including incident threads.

Note The XML API incident_get function does not return data fields with NULL values.

incident_update

The incident_update function is used to update the information associated with an existing incident in the RightNow CRM database. The function can have three components: the i_id of the incident as an integer, an array of pair data, and logical flag information. A valid i_id of an existing incident must be supplied in the parameter tag. If the function executes without error, a 1 will be returned. If no valid i_id is supplied, or no incident exists with the supplied ID number, a zero will be returned.

The API will set any fields supplied in the pair list, including custom fields and incident threads. Any incident fields missing from the pair list will not be altered in the database.

Example:

```
<connector>
     <function name="incident_update">
          <parameter name="i_id">7</parameter>
          <parameter name="args">
               <pair name="status_id" type="integer">2</pair>
               <pair name="assgn_acct_id" type="integer">4</pair>
               <pair name="custom_field" type="pair">
                    <pair name="cf_item" type="pair">
```

```
                                <pair name="cf_id" type="integer">4</pair>
                                <pair name="value">Brakes</pair>
                          </pair>
                    </pair>
              </parameter>
        </function>
  </connector>
```

This example updates the incident with the ID of 7 to assign the incident to the staff member with ID number 4. The status is set to solved (status code 2) and a custom field (code 4) is set to the value "Brakes."

Meta-answer API

The meta-answer API functions (meta_ans_create, meta_ans_destroy, and meta_ans_update) allow you to create or alter information from the *meta_answers* table. You can act on all standard database fields of the *meta_answers* table.

Meta-answers are groups of answers that solve the same question, but present information in different formats, either in another language or at different levels of detail. The meta-answer defines the summary line of the answer it is associated with, as well as products and categories assigned to that answer. However, you must edit the answer separately to define content and visibility settings.

Because meta-answers can be associated with multiple products or categories, you must use a unique pair structure to define these when using the meta_ans_create or meta_ans_update functions. Refer to the examples in this section for specific instructions on setting multiple products or categories.

meta_ans_create

The meta_ans_create function is used to add a meta-answer to the RightNow database. The function has one component: an array of pair data.

Important The API will automatically generate a meta-answer ID for the meta-answer that is consistent with existing meta-answers in the database.

The meta-answer will be populated with data specified in the pair list. A brief description of all *meta_answers* table fields and their corresponding pair names can be found in Appendix A, "Pair Names," on page 103.

You must include a title for the meta-answer when using this function. Products and categories are also required if enabled in your configuration.

Note You must create an answer to associate with the meta-answer for the meta-answer to appear on the administration interface. Refer to “ans_create” on page 32.

You can assign multiple products and categories to a meta-answer by using multiple hier_item pairs within a products or categories pair array. You can create six levels of products and categories using this method.

Example:

```
<connector>
     <function name="meta_ans_create">
          <parameter name="args" type="pair">
               <pair name="summary" type="string">How do I access
               voicemail for my mobile phone?</pair>
               <pair name="products" type="pair">
                    <pair name="hier_item" type="pair">
                         <pair name="id1" type="integer">1</pair>
                    </pair>
                    <pair name="hier_item" type="pair">
                         <pair name="id1" type="integer">3</pair>
                         <pair name="id2" type="integer">12</pair>
                    </pair>
                    <pair name="hier_item" type="pair">
                         <pair name="id1" type="integer">5</pair>
                         <pair name="id2" type="integer">22</pair>
                         <pair name="id3" type="integer">33</pair>
                    </pair>
               </pair>
               <pair name="categories" type="pair">
                    <pair name="hier_item" type="pair">
                         <pair name="id1" type="integer">13</pair>
                    </pair>
                    <pair name="hier_item" type="pair">
                         <pair name="id1" type="integer">15</pair>
                         <pair name="id2" type="integer">42</pair>
                    </pair>
                    <pair name="hier_item" type="pair">
                         <pair name="id1" type="integer">18</pair>
```

```
                            <pair name="id2" type="integer">45</pair>
                            <pair name="id3" type="integer">51</pair>
                       </pair>
                  </pair>
             </parameter>
        </function>
</connector>
```

In this example, a meta-answer with the title "How do I access voicemail for my mobile phone?" is created in RightNow CRM. The meta-answer is associated with three first-level products (IDs of 1, 3, and 5), two second-level products (IDs of 12 and 22), and one level-three product (ID of 33). The meta-answer is also associated with a first-level category (IDs of 13, 15, and 18), a second-level category (IDs of 42 and 45), and a third-level category (ID of 51). The value of the m_id is automatically created and returned from the database.

Note When defining products and categories with multiple levels, the ID pairs specified within the hier_item pair must match the hierarchy of your products and categories. For example, in the previous example, the category with ID 51 must be a lower-level category that is associated with category with ID 45 which must be a lower-level category associated with the category with ID 18.

meta_ans_destroy

The meta_ans_destroy function is used to delete an existing meta-answer in the RightNow database. The function has one component: the m_id of the meta-answer. A valid m_id of an existing meta-answer must be supplied. If the function executes without error, a 1 will be returned. If the m_id is not set, or no meta-answer exists with the supplied ID number, the function will abort with an error message. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

Note Deleting a meta-answer will delete all answers associated with it.

Example:

```
<connector>
     <function name="meta_ans_destroy">
          <parameter name="m_id" type="integer">25</parameter>
     </function>
</connector>
```

This example deletes the meta-answer with an ID number of 25 from the database.

meta_ans_update

The meta_ans_update function is used to update the information associated with an existing meta-answer in the RightNow database. The function has two components: the m_id of the meta-answer and an array of pair data. A valid m_id of an existing meta-answer must be supplied in the parameter tag. If the function executes without error, a 1 will be returned. If the meta-answer ID is not set, or no meta-answer exists with the supplied meta-answer ID, the function will abort with an error. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

The API will set any fields supplied in the pair list, and any meta-answer fields missing from the pair list will not be altered in the database.

Example:

```
<connector>
     <function name="meta_ans_update">
          <parameter name="m_id" type="integer">25</parameter>
          <parameter name="args" type="pair">
               <pair name="categories" type="pair">
                    <pair name="hier_item" type="pair">
                         <pair name="id1" type="integer">18</pair>
                    </pair>
                    <pair name="hier_item" type="pair">
                         <pair name="id1" type="integer">13</pair>
                         <pair name="id2" type="integer">42</pair>
                    </pair>
                    <pair name="hier_item" type="pair">
                         <pair name="id1" type="integer">15</pair>
                         <pair name="id2" type="integer">45</pair>
                         <pair name="id3" type="integer">51</pair>
                    </pair>
               </pair>
          </parameter>
     </function>
</connector>
```

This example updates the meta-answer with an meta-answer ID of 25 to specify three additional category associations.

Opportunity API

The opportunity API functions (sa_opp_create, sa_opp_destroy, sa_opp_get, and sa_opp_update) allow you to create, delete, retrieve, or update information from the *sa_opportunities* table. You can act on all standard database fields of the *sa_opportunities* table, as well as some specialized information, such as custom fields.

sa_opp_create

The sa_opp_create function is used to add an opportunity to the RightNow database. The function has two components: an array of pair data and logical flag information.

Important The API will automatically generate an op_id for the opportunity that is consistent with existing opportunities in the database.

The opportunity will be populated with data specified in the pair list. A brief description of all opportunities table fields and their corresponding pair names can be found in Appendix A, "Pair Names," on page 103. A brief description of all opportunity source codes can be found in Appendix B, "Source Codes," on page 127.

Note Thread entries in opportunities use a unique pair structure. For more information, refer to "Thread entry types" on page 72.

You must provide a name, summary, status, organization ID, and a primary contact for the opportunity in your XML. To create a hierarchy of territories or sales representatives, use the terr_lvl<1-12>_id and acct_lvl<1-11>_id pairs. The top-level territory is specified using terr_lvl1_id, the second-level territory is specified using terr_lvl2_id, and so on. The territory hierarchy in the opportunity must match the territory hierarchy in the Management and Configuration Console.

Important To set the sales representative, you must use the assgn_acct_id pair, along with the acct_lvl<1-11>_id pair. The top-level sales representative, or manager, is specified using acct_lvl1_id, the second-level sales representative is specified using acct_lvl2_id, and so on. The account hierarchy in the opportunity must match the account hierarchy in the Management and Configuration Console.

You can assign multiple secondary contacts to an opportunity using the opp2contact pair along with an array of pair data. Separate contacts are specified using the oc_item<#> pair. The first contact is specified by oc_item0, the second contact is specified by oc_item1, and so on. You must set a contact role for each contact and indicate whether it is the primary contact.

Example:

```
<connector>
     <function name="sa_opp_create" id="opp2">
          <parameter name="args" type="pair">
               <pair name="updated_by" type="integer">2</pair>
               <pair name="source" type="integer">21</pair>
               <pair name="status_id" type="integer">9</pair>
               <pair name="opp2contact" type="pair">
                    <pair name="oc_item0" type="pair">
                         <pair name="c_id" type="integer">1</pair>
                         <pair name="cr_id" type="integer">1</pair>
                         <pair name="oc_primary" type="integer">1</pair>
                    </pair>
                    <pair name="oc_item1" type="pair">
                         <pair name="c_id" type="integer">3</pair>
                         <pair name="cr_id" type="integer">2</pair>
                         <pair name="oc_primary" type="integer">0</pair>
                    </pair>
               </pair>
               <pair name="org_id" type="integer">1</pair>
               <pair name="name" type="string">Fall Clearance Sale</pair>
               <pair name="summary" type="string">40% off on all camera
phones for existing customers.</pair>
          </parameter>
     </function>
</connector>
```

This example creates an opportunity, Fall Clearance Sale, and associates two contacts with the opportunity. This function will automatically return the opportunity ID from the database.

sa_opp_destroy

The sa_opp_destroy function is used to delete an existing opportunity in the RightNow database. The function has one component: the op_id of the opportunity. A valid op_id of an existing opportunity must be supplied. If the function executes without error, a 1 will be returned. If the op_id is not set, or no opportunity exists with the supplied ID number, the function will abort with an error message. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

Example:

```
<connector>
     <function name="sa_opp_destroy">
```

```
          <parameter name="op_id" type="integer">8</parameter>
     </function>
</connector>
```

This example deletes the opportunity with the ID number 8 from the database.

sa_opp_get

The sa_opp_get function is used to retrieve a record from the *sa_opportunities* table. The single component of this function is the op_id. A valid ID number of an existing opportunity must be supplied. If no valid op_id is supplied, a blank value will be returned.

Example:

```
<connector>
     <function name="sa_opp_get">
          <parameter name="op_id" type="integer">9</parameter>
          <parameter name="flags" type="integer">0x00200</parameter>
     </function>
</connector>
```

This example retrieves the opportunity details with ID number 9 from the database. All opportunity API pairs are returned using this function, including opportunity note threads.

Note The XML API sa_opp_get function does not return data fields with NULL values.

sa_opp_update

The sa_opp_update function is used to update the information associated with an existing opportunity in the RightNow database. The function has three components: the op_id, an array of pair data, and logical flag information. A valid opportunity ID of an existing opportunity must be supplied in a parameter tag. If the function executes without error, a 1 will be returned. If the op_id is not set, or no opportunity exists with the supplied opportunity ID, the function will abort with an error. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

The API will set any fields supplied in the pair list, including custom fields. Any opportunity fields missing from the pair list will not be altered in the database.

Note If another contact is added or one of the contacts is deleted from the contact list, the entire new list needs to be passed because the API deletes the existing contact list and inserts the new list if there is any change. If there is no change, opp2contact does not need to be set on update.

Example:

```
<connector>
     <function name="sa_opp_update">
          <parameter name="op_id" type="integer">9</parameter>
          <parameter name="args" type="pair">
               <pair name="status_id" type="integer">11</pair>
          </parameter>
     </function>
</connector>
```

This example updates the opportunity with ID number 9. The opportunity's status is changed to an ID of 11 (Closed).

Organization API

The organization API functions (org_create, org_destroy, org_get, and org_update) allow you to create, delete, retrieve, or update information from the *orgs* table. You can act on all standard database fields of the *orgs* table, as well as some specialized information, such as custom fields.

Contacts can be associated with organizations in RightNow CRM. By associating contacts with organizations, contacts and staff members can view all incidents submitted by an organization and allow administrators to assign an SLA instance to all contacts in an organization.

An organization can have several types of addresses, including a billing and shipping address. When passing address information using the org_create or org_update function, a unique pair structure is used. Table 5 describes the default address types that can be associated with each organization.

Table 5: Address Type Descriptions

Address Type (oat_id)	**ID**
Billing	1
Shipping	2

The following example shows the pair for each billing and shipping.

Example:

```
<connector>
```

```
     <function name="org_update">
          <parameter name="org_id" type="integer">27</parameter>
          <parameter name="args" type="pair">
               <pair name="oaddr" type="pair">
                    <pair name="oaddr_item" type="pair">
                         <pair name="oat_id" type="integer">1</pair>
                         <pair name="street" type="string">12345 Maple
                         Way</pair>
                         <pair name="city" type="string">Bozeman</pair>
                         <pair name="prov_id" type="integer">32</pair>
                         <pair name="postal_code" type="string">59715
                         </pair>
                         <pair name="country_id" type="integer">1</pair>
                    </pair>
                    <pair name="oaddr_item" type="pair">
                         <pair name="oat_id" type="integer">2</pair>
                         <pair name="street" type="string">321 Oak Street
                         </pair>
                         <pair name="city" type="string">Belgrade</pair>
                         <pair name="prov_id" type="integer">32</pair>
                         <pair name="postal_code" type="string">59714
                         </pair>
                         <pair name="country_id" type="integer">1</pair>
                    </pair>
               </pair>
          </parameter>
     </function>
</connector>
```

Note For sales contacts, thread entries can be added or updated for contact notes, email, phone and notes field. These fields appear on the Notes tab in the contact record. For more information refer to "Thread entry types" on page 72.

org_create

The org_create function is used to add an organization to the RightNow database. The function can have two components: an array of pair data and logical flag information. An organization name and login must be supplied.

Important The API will automatically generate an org_id for the organization that is consistent with existing organizations in the database. The CRM state will be set to Service by default.

The organization will be populated with data specified in the pair list. A brief description of all *orgs* table fields and their corresponding pair names can be found in Appendix A, "Pair Names," on page 103. A brief description of all organization source codes can be found in Appendix B, "Source Codes," on page 127.

Example:

```
<connector>
     <function name="org_create">
          <parameter name="args" type="pair">
               <pair name="name" type="string">The River Deep</pair>
               <pair name="login" type="string">riverdeep</pair>
               <pair name="password" type="string">wh1tewat3r</pair>
          </parameter>
          <parameter name="flags" type="integer">0x00002</parameter>
     </function>
</connector>
```

This example creates an organization, The River Deep, with a login of "riverdeep" and a password of "wh1tewat3r." The function will return the organization ID number and execute an external event.

org_destroy

The org_destroy function is used to delete an existing organization in the RightNow database. The function can have two components: the ID number of the organization and logical flag information. A valid org_id of an existing organization must be supplied. If the function executes without error, a 1 will be returned. If the org_id is not set, or no organization exists with the supplied ID number, the function will abort with an error message. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

Caution Deleting an organization will result in the deletion of all contacts, incidents, and opportunities associated with the organization.

Example:

```
<connector>
     <function name="org_destroy">
          <parameter name="org_id" type="integer">8</parameter>
          <parameter name="flags" type="integer">0x00002</parameter>
     </function>
</connector>
```

This example deletes the organization with ID number 8 and executes an external event.

org_get

The org_get function is used to retrieve a record from the *orgs* table. The single component of this function is the organization ID. A valid org_id of an existing organization must be supplied. If no valid org_id is supplied, a blank value will be returned.

Example:

```
<connector>
     <function name="org_get">
          <parameter name="org_id" type="integer">7</parameter>
     </function>
</connector>
```

This example retrieves the organization details with ID number 7 from the database.

Note The XML API org_get function does not return data fields with NULL values.

org_update

The org_update function is used to update the information associated with an existing organization in the RightNow database. The function can have three components: the org_id of the organization, an array of pair data, and logical flag information. A valid org_id of an existing organization must be supplied in the parameter tag. If the function executes without error, a 1 will be returned. If the org_id is not set, or no organization exists with the supplied org_id, the function will abort with an error. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

The API will set any fields supplied in the pair list, including custom fields. Any organization fields missing from the pair list will not be altered in the database.

Example:

```
<connector>
     <function name="org_update">
          <parameter name="org_id" type="integer">9</parameter>
          <parameter name="args" type="pair">
               <pair name="password" type="string">newpassword</pair>
          </parameter>
     </function>
</connector>
```

This example changes the password of the organization with ID number 9 to "newpassword."

Quote API

The quote API functions (sa_quote_destroy, sa_quote_get, and sa_quote_update) allow you to delete, retrieve, or update information from the *sa_quotes* table. You can act on all standard database fields of the *sa_quotes* table, as well as some specialized information, such as custom fields.

sa_quote_destroy

The sa_quote_destroy function is used to delete an existing quote in the RightNow database. The function has one component: the quote_id of the quote. A valid quote_id of an existing quote must be supplied. If the function executes without error, a 1 will be returned. If the quote_id is not set, or no quote exists with the supplied ID number, the function will abort with an error message. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

Example:

```
<connector>
     <function name="sa_quote_destroy">
          <parameter name="quote_id" type="integer">82</parameter>
     </function>
<connector>
```

This example deletes the quote with the ID number 82 from the database.

sa_quote_get

The sa_quote_get function is used to retrieve a record from the *sa_quotes* table. The single component of this function is the quote_id. A valid ID number of an existing quote must be supplied. If no valid quote_id is supplied, a blank value will be returned.

Example:

```
<connector>
     <function name="sa_quote_get">
          <parameter name="quote_id" type="integer">1</parameter>
     </function>
</connector>
```

This example retrieves the quote details with ID number 1 from the database.

Note The XML API sa_quote_get function does not return data fields with NULL values.

sa_quote_update

The sa_quote_update function is used to update the information associated with an existing quote in the RightNow database. The function has two components: the quote_id and array of pair data. A valid quote ID of an existing quote must be supplied in a parameter tag. If the function executes without error, a 1 will be returned. If no valid quote_id is supplied, a 0 will be returned.

The API will set any fields supplied in the pair list, including custom fields. Any quote fields missing from the pair list will not be altered in the database.

Example:

```
<connector>
     <function name="sa_quote_update">
          <parameter name="quote_id" type="integer">9</parameter>
          <parameter name="args" type="pair">
               <pair name="status" type="integer">4</pair>
          </parameter>
     </function>
</connector>
```

This example updates the quote with ID number 9. The quote's status is changed to an ID of 4 (Returned).

Search API

The XML search API function can be used to search for records in RightNow CRM, including incidents, answers, meta-answers, contacts, organizations, opportunities, quotes, and tasks.

To perform a search using the API, you must provide a valid view ID and search fields. All fixed filters defined in the view are applied to the query that is run by the XML search API function and any run-time filters with default values are also included. To narrow your search, you can pass the run-time selectable filters created in the view.

Note The view ID must be defined on the current interface to work correctly.

The run-time selectable filters are passed in a search_args pair. To identify the filter you want to set, you use a name pair and set the name to the name of the filter (the name is specified by you when you create the filter in the view). The value is passed using the compare_value pair. The format of the run-time selectable filters depends on the type of connector used in the filter. Refer to the following table for a description of each type of connector.

Note If your view output data length is set to more than 4000 characters, the XML search API will truncate the return results at the 4000 character limit.

Table 6: View Operator Descriptions

Operator	Description

=, !=, <, <=, >, >=, like, not like, is null, != or null, not like or null	The value should be a number or string.
	For example, to search for a particular subject, you would use the following:
	`<pair name="compare_value" type="string">`
	`maintenance</pair>`

Table 6: View Operator Descriptions (Continued)

Operator	Description

in list, not in list	The value should be a list of numbers separated by semicolons.
	For example, to search for two statuses (IDs are 4 and 5), you would use the following:
	`<pair name="compare_value" type="string">4;5 </pair>`
	
	When searching for products and categories, you must specify the level the code ID is associated with. The format is <level>.<ID>. For example, to search for a product (ID is 2) and two of its lower-level products (IDs are 9 and 12), you would use the following:
	`<pair name="compare_value" type="string"> 1.2;2.9;2.12</pair>`
	
	To search for something that has product = 5 and sub-level NULL, you specify “2.u5”, which says that the level two ID should be NULL and the level 1 ID should be 5. You can combine this with others as follows:
	`<pair name="compare_value" type="string"> 1.2;2.u5</pair>`
	In the example listed above, “1.2;2.u5” would equate to “prod_lvl1 = 2 OR (prod_lvl1 = 5 AND prod_lvl2 IS NULL).”
	
	If you want to specify that the product should be NULL, you use “1.u0”, which is a special case, since the level 1 values have no parents.
	
	Note: Six levels of products and categories are supported. For example, “1.2;3.22;4.u35” would search everything with “prod_lvl1 = 2 or prod_lvl3=22 or (prod_lvl3=35 and prod_lvl4 is null).”
	
	In the previous example, the “4.u35” describes that prod_lvl4 should equal something. In this particular case, the “u35” describes that prod_lvl4 should be null, but the parent should be 35 (which means prod_lvl3=35).
	
	In other words, if the string was “1.9;4.u23”, it would expand to prod_lvl1=9 OR (prod_lvl4 is NULL and prod_lvl3=23).

Table 6: View Operator Descriptions (Continued)

Operator	Description
between	The value should be two numbers, separated by a pipe (|). For example, to search for answers with an ID between 1 and 50, you would use the following: `<pair name="compare_value" type="string">1|50 </pair>`

You can use the max_rows parameter to pass the maximum number of rows returned by the search. If a value is not specified for this parameter, the value in the configuration setting VRL_SOFT is used. The upper limit of the allowed number passed in this parameter is set by the configuration setting VRL_HARD.

Examples:

The following example produces a default result set defined by the referenced answer view ID.

```
<connector>
     <function name="search">
          <parameter name="view_id" type="integer">10</parameter>
     </function>
</connector>
```

The following example shows a search by product and a range of answer IDs.

```
<connector>
     <function name="search">
          <parameter name="args" type="pair">
               <pair name="search_args" type="pair">
                    <pair name="search_field1" type="pair">
                         <pair name="name" type="string">product</pair>
                         <pair name="compare_value" type="string">
                         1.2;2.9;2.12;</pair>
                    </pair>
                    <pair name="search_field2" type="pair">
                         <pair name="name" type="string">a_id</pair>
                         <pair name="compare_value" type="string">1|50
                         </pair>
                    </pair>
               </pair>
          </parameter>
          <parameter name="view_id" type="integer">2</parameter>
          <parameter name="max_rows" type="integer">5</parameter
     </function>
</connector>
```

In this example, the function searches for all answers with a product ID of 2 and a lower-level product ID of 9 or 12. It also searches for answers with an ID between the range of 1 and 50 and returns the values according to the default Answer Console view (2).

Example result set:

The following is an example of a set of results from an answer search.

```
<connector_ret>
<function name="search" id="">
     <row id="1">
          <col id="1">1</col>
          <col id="2">How do I email a photo with my camera phone?
          </col>
          <col id="3">en_US</col>
          <col id="4">Everyone</col>
          <col id="5">Public</col>
          <col id="6">Mary Smith</col>
          <col id="7">1036594069</col>
     </row>
     <row id="2">
          <col id="1">2</col>
          <col id="2">What will it cost for me to upgrade to your business
          plan?</col>
          <col id="3">en_US</col>
          <col id="4">Everyone</col>
          <col id="5">Public</col>
          <col id="6">Mary Smith</col>
          <col id="7">1036594069</col>
     </row>
</function>
</connector_ret>
```

The above result set shows a search return value containing two rows, or two matching records for a search. Each row relates directly to a row in the specified view. To find the *view_id* for a desired view, refer to "Finding code numbers" on page 75.

Caution Changing your RightNow CRM architecture may alter or adversely affect your search results.

SLA instance API

The SLA API functions (slai_create and slai_terminate) allows you to create or delete an SLA instance within the *sla_instances* table. You can act on all standard database fields of the *sla_instances* table.

An SLA instance is an assigned SLA that agents can assign from the Support Console or through the XML API and associate with a contact or organization that has already been assigned an sla_id. For additional information on SLAs, see the *RightNow Service 7.0 Administration Manual.*

slai_create

The slai_create function is used to add an SLA instance to the RightNow database. The function has one component: an array of pair data that includes owner_tbl, owner_id, sla_id, and activedate.

Important The API will automatically generate an slai_id for the SLA instance that is consistent with existing SLA instances in the database.

The SLA instance will be populated with data specified in the pair list. A brief description of all *sla_instances* table fields and their corresponding pair names can be found in Appendix A, "Pair Names," on page 103.

Example:

```
<connector>
     <function name="slai_create">
          <parameter name="args" type="pair">
               <pair name="activedate" type="integer">1084406905</pair>
               <pair name="owner_id" type="integer">1</pair>
               <pair name="owner_tbl" type="integer">3</pair>
               <pair name="sla_id" type="integer">1</pair>
          </parameter>
     </function>
</connector>
```

This example creates an SLA instance in the *sla_instances* table, sets the owner_id to 1, the owner_tbl to 3 (the *orgs* table), and automatically returns the slai_id. The owner_id corresponds with the c_id of the contact or org_id of the organization the SLA is associated with. The owner_tbl corresponds with the table ID of the table the owner_id is associated with; contact-associated SLAs will have a table ID of 2 and organization-associated SLAs will have a table ID of 3.

slai_terminate

The slai_terminate function is used to delete an existing SLA instance in the RightNow database. The function has one component: the slai_id. A valid slai_id of an existing SLA instance must be supplied. If the slai_id is not set, or no SLA instance exists with the supplied ID number, the function will abort with an error message. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

Example:

```
<connector>
     <function name="slai_terminate">
          <parameter name="slai_id" type="integer">10</parameter>
     </function>
</connector>
```

This example deletes the SLA instance with ID 10 from the database.

SQL query API

The SQL query API functions (sql_get_int, sql_get_str, sql_get_time) allow read-only access to the RightNow database through the XML API. These functions will return a single value from the database. The ID attribute and the sql parameter must be supplied for these functions.

When using any of the sql_get functions, if more than one value meets the criteria of the SQL statement, only the first value to match the criteria will be returned. For this reason, you should not use an SQL statement like `SELECT * from <table>` because only the first value in the table will be returned; however, a SQL statements like `SELECT COUNT(*) from <table>` or `SELECT MAX(acct_id) FROM accounts` would work well because they only return a single value.

Note The terminating semicolon is implied for all SQL statements.

sql_get_int

The sql_get_int function is used to execute a SELECT statement against the RightNow database when the result is an integer, such as an account ID from the *accounts* table or a count of records in the *incidents* table. The function has one component: the sql parameter. A single integer will be returned.

Example:

```
   <connector>
```

```
    <function name="sql_get_int" id="sql_str">
         <parameter name="sql" type="string">
              SELECT acct_id FROM accounts WHERE login = 'susan'
         </parameter>
    </function>
  </connector>
```

Return:

```
  <?xml version="1.0" encoding="UTF-8" ?>
    <connector_ret>
         <function name="sql_get_int" id="sql_str">
              <ret_val name="rv" type="integer">8</ret_val>
         </function>
  </connector_ret>
```

This example runs an SQL query to find the account ID for the account with a login of "susan" and returns the integer "8."

sql_get_str

The sql_get_str function is used to execute a SELECT statement against the RightNow database when the result is a string, such as the login name from the *accounts* table. The function has one component: the sql parameter. A single string will be returned.

Example:

```
  <connector>
    <function name="sql_get_str" id="sql_str">
         <parameter name="sql" type="string">
              SELECT login FROM accounts WHERE acct_id = 10
         </parameter>
    </function>
  </connector>
```

Return:

```
  <?xml version="1.0" encoding="UTF-8" ?>
  <connector_ret>
    <function name="sql_get_str" id="sql_str">
         <ret_val name="rv" type="string">archie</ret_val>
    </function>
  </connector_ret>
```

This example runs an SQL query to find the account login for the account with the ID of 10 and returns the string "archie."

sql_get_time

The sql_get_time function is used to execute a SELECT statement against the RightNow database when the result is a timestamp, such as the password expiration time from the *accounts* table. The function has one component: the sql parameter. A single timestamp will be returned in UNIX date_t format (the number of seconds since the UNIX Epoch date).

Example:

```
<connector>
  <function name="sql_get_time">
       <parameter name="sql" type="string">
            SELECT password_exp FROM accounts WHERE acct_id = 10
       </parameter>
  </function>
</connector>
```

Return:

```
<?xml version="1.0" encoding="UTF-8" ?>
<connector_ret>
  <function name="sql_get_time">
       <ret_val name="rv" type="time">1093330800</ret_val>
  </function>
</connector_ret>
```

This example runs an SQL query to find the password expiration time for the account with the ID of 10 and returns the value "1093330800."

Task instance API

The task instance API functions (sa_task_ins_create, sa_task_ins_destroy, sa_task_ins_get, and sa_task_ins_update) relate to the Sales Console and allow you to create, update, delete, or retrieve a task instance from the *sa_task_instances* table.

sa_task_ins_create

The sa_task_ins_create function is used to add a task instance to the RightNow database. The function has one component: an array of pair data.

Important The API will automatically generate a ti_id for the task instance. If no name is specified for the task instance, it will be named New Task. If no source is specified for the task instance, it will be set to 10 to indicate it was created through the API.

The task will be populated with data specified in the pair list. A brief description of all task instance table fields and their corresponding pair names can be found in Appendix A, “Pair Names,” on page 103.

Example:

```
<connector>
     <function name="sa_task_ins_create">
          <parameter name="args" type="pair">
               <pair name="name" type="string">Schedule Executive Call
               </pair>
               <pair name="source" type="integer">10</pair>
          </parameter>
     </function>
</connector>
```

This example creates a task instance with the name “Task Instance from API.” The API will automatically generate a ti_id.

sa_task_ins_destroy

The sa_task_ins_destroy function is used to delete an existing task instance in the RightNow database. The function has one component: the ti_id of the task instance. A valid ti_id of an existing task instance must be supplied. If the function executes without error, a 1 will be returned. If the ti_id is not set, or no task instance exists with the supplied ID number, the function will abort with an error message. A zero will be returned if the ID number is invalid or -1 will be returned if an error occurs.

Example:

```
<connector>
     <function name="sa_task_ins_destroy">
          <parameter name="ti_id" type="integer">7</parameter>
     </function>
</connector>
```

This example deletes the task instance with the ID number 7 from the database.

sa_task_ins_get

The sa_task_ins_get function is used to retrieve a record from the sa_task_instances table. The single component of this function is the ti_id. A valid ID number of an existing task instance must be supplied. If the function executes without error, the task instance details will be returned. If no valid ti_id is supplied, a blank value will be returned.

Example:

```
<connector>
     <function name="sa_task_ins_get">
          <parameter name="ti_id" type="integer">7</parameter>
     </function>
</connector>
```

This example retrieves the task instance details with ID number 7 from the database.

Note The XML API sa_task_ins_get function does not return data fields with NULL values.

sa_task_ins_update

The sa_task_ins_update function is used to update the information associated with an existing task instance in the RightNow database. The function has two components: the ti_id and an array of pair data.

The API will set any fields supplied in the pair list, including custom fields. Any task instance fields missing from the pair list will not be altered in the database. If the function executes without error, a 1 will be returned. If no valid ti_id is supplied, a 0 will be returned.

Example:

```
<connector>
     <function name="sa_task_ins_update">
          <parameter name="ti_id" type="integer">7</parameter>
          <parameter name="args" type="pair">
               <pair name="name" type="string">Updated Task Instance from
               API</pair>
               <pair name="source" type="integer">10</pair>
          </parameter>
     </function>
</connector>
```

This example updates the task instance name to “Updated Task Instance from API.”

Implementing code for the XML API

The following sections describe the two methods you can use to implement code for use with the RightNow XML API, along with some tips for passing thread entry types and variable IDs, looking up code numbers, and using the XML API log.

Using the POST method

When using the POST method, the XML is immediately sent to RightNow CRM and parsed by the PHP script (*parse.php*). Record data is then instantly created, updated, or deleted in the RightNow database. The *parse.php* script is located at:

```
http://<your_domain>/cgi-bin/<your_interface>.cfg/php/xml_api/parse.php
```

To develop the integration, you will need to create code operating independently or within the HTML on a separate web page to post the XML data. The posted data must pass two parameters: xml_doc and sec_string. The xml_doc parameter contains the entire set of XML data, including the <connector> and </connector> tags and all XML contained within the tags. The sec_string parameter should specify the XML trigger phrase specified in the II_SEC_STRING configuration setting (refer to Table 7 on page 70).

Note The encoding of *parse.php* is set to UTF-8, and any XML document passed to the parser must also be UTF-8 encoded.

A simple way to use the POST method to send XML to RightNow CRM is to create a web form using HTML, as shown in the following example:

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<form method="POST" action="http://<your_domain>/cgi-bin/
<your_interface>.cfg/php/xml_api/parse.php" name="XML Form">
     <h2>XML Data</h2>
     <textarea cols="80" name="xml_doc" rows="20"></textarea>
          <br><br>
     <h2>Security String</h2>
     <input name="sec_string" size="10" value = "xml">
          <br><br>
     <input type="submit" value="Submit" name="B1">
     <input type="reset" value="Reset" name="B2">
</form>
```

In this example, a web form with two text boxes (for XML data and to pass the security string) is created. The security string text box is prepopulated with the value specified in the II_SEC_STRING configuration setting (in this case, "xml"). You can then use this web page to enter XML data and submit it to *parse.php*.

You can also post XML data to RightNow CRM using this method but without using a web page, by directly opening a socket connection to *parse.php*. You can accomplish this using any scripting language, such as PHP. Using this method, you must establish a connection with RightNow CRM, and then use POST to pass your XML data.

RightNow Technologies Professional Services can assist you in determining which XML integration method best suits your needs and then implementing the method. For more information, contact your RightNow account manager.

Sending an XML-formatted email

You can add, update, delete, and retrieve data or perform a search or lookup function through the RightNow CRM API by sending an XML-formatted email to a RightNow CRM mailbox. The email must have a trigger word or phrase in the subject line that is specified in the RightNow CRM configuration settings. When RightNow CRM receives the email, the utility *techmail* will identify it as XML through the trigger word or phrase. The email will then be parsed by a PHP script to retrieve the data.

You must configure RightNow CRM to identify email that contains data in XML format. Through the configuration setting II_SEC_STRING, you can specify a value for a trigger phrase to be used in the subject line of the email. The value specified by this configuration setting must be matched exactly, including case, to identify the email as XML. You can also configure RightNow CRM to send an email message if there are any errors during the XML integration.

The configuration settings are located under RNT Common>External Events>Incoming Integration and are detailed in Table 7.

Table 7: Incoming Integration Configuration Settings

Configuration Setting	Description
II_EMAIL_ERROR_ADDR	Specifies the email address to send XML API error data. Default is blank.

Table 7: Incoming Integration Configuration Settings (Continued)

Configuration Setting	Description
II_SEC_STRING	Specifies the post variable from a web page or the subject line of email to be compared for validation of the XML source. This is used to provide an interface to the RightNow XML API for third-party call management systems or other third-party systems. Default is blank.

Important XML-formatted email messages must be in plain text.

Setting custom fields

Passing custom field data through the API is different than interacting with standard database fields. To set a custom field using a create or update function, each custom field must be specified in its own <pair>. The pair name should be set to custom_field with a type of pair, with an embedded pair name set to cf_item with a type of pair. This pair contains two additional sets of pairs.

The first pair specifies the code of the custom field. In this pair, the name should be set to "cf_id" with a type of "integer," and the value of the pair will be the code number of the custom field.

The second pair specifies the value you want the custom field set to. In this pair, the name should be set to "value." For menu custom fields, this pair should be a string type with the value of the pair set to the code number of the menu item. Refer to "Finding code numbers" on page 75 for information about finding code numbers.

Examples:

The following example shows how to set a text field-type custom field, where the custom field with code 4 is set to "Brakes."

```
<connector>
     <function name=”incident_update”>
          <parameter name="i_id" type="integer">726</parameter>
          <parameter name="args" type="pair">
               <pair name="custom_field" type="pair">
                    <pair name="cf_item" type="pair">
                         <pair name="cf_id" type="integer">4</pair>
                         <pair name="value" type="string">Brakes</pair>
                    </pair>
          </pair>
     </parameter>
</function>
</pair>
```

The following example shows how to set a radio button-type custom field. The “value” pair is set to 1 for “yes,” or 0 for “no.”

```
<connector>
     <function name=”incident_update”>
          <parameter name="i_id" type="integer">964</parameter>
          <parameter name="args" type="pair">
               <pair name="custom_field" type="pair">
                    <pair name="cf_item" type="pair">
                         <pair name="cf_id" type="integer">12</pair>
                         <pair name="value" type="integer">1</pair>
                    </pair>
               </pair>
          </parameter>
     </function>
</connector>
```

Note Date and date/time custom fields must be configured with a type of “time.” The value must be in UNIX date_t format; that is, a long integer that is the number of seconds since the UNIX Epoch date (00:00:00 UTC January 1, 1970).

Thread entry types

An incident can contain a threaded conversation between staff members and end-users. A sales contact, sales opportunity, or sales organization can contain threaded entries by staff members only. Creating threads with the XML API create and update functions has a different structure than other pairs. A thread is created as a pair structure, which allows you to specify the type of thread that is associated with the incident, contact, organization, or opportunity. The following table describes the thread types that can be associated with each record.

Table 8: Thread Entry Type Descriptions

Thread Entry Type	ID
Note	1
Staff	2

Table 8: Thread Entry Type Descriptions (Continued)

Thread Entry Type	ID
Contact	3
Contact Proxy	4
RightNow Live	5
Rule Response	6
Rule Response Note	7
Sales Note	8
Sales Customer Email	9
Sales Email	10
Sales Phone	11

The following incident_update example shows a thread pair:

```
<connector>
     <function name="incident_update">
          <parameter name="i_id" type="integer">7</parameter>
          <parameter name="args" type="pair">
               <pair name="thread" type="pair">
                    <pair name="thread_entry" type="pair">
                         <pair name="entry_type" type="integer">3</pair>
                         <pair name="note" type="string">How do I access
                         voice mail?
                         </pair>
                    </pair>
               </pair>
          </parameter>
     </function>
</connector>
```

Passing variable IDs

When you use multiple XML functions in the same XML file, the XML API allows you to store newly created record IDs in a variable to be used later in your XML. To create a variable, define the variable using the id attribute in the function tag as shown in the following example.

```
<function name="org_create" id="organization_id">
```

In this example, the org_id assigned to the new organization will be stored in the variable organization_id. This variable can be called later in your XML by replacing the org_id with the variable $organization_id.

The following example shows how you can create a contact and also create an incident associated with that contact in the same XML file by creating and passing the variable contact_id.

Example:

```
<connector>
     <function name="contact_create" id="contact_id">
          <parameter name="args" type="pair">
               <pair name="login" type="string">jsmith</pair>
               <pair name="first_name" type="string">Joe</pair>
               <pair name="last_name" type="string">Smith</pair>
               <pair name="email" type="string">jsmith@example.com</pair>
          </parameter>
     </function>
     <function name="incident_create">
          <parameter name="args" type="pair">
               <pair name="c_id" type="integer">$contact_id</pair>
               <pair name="subject" type="string">What are Free Minutes?
               </pair>
               <pair name="thread" type="pair">
                    <pair name="thread_entry" type="pair">
                         <pair name="entry_type" type="integer">3</pair>
                         <pair name="c_id" type="integer">$contact_id
                         </pair>
                         <pair name="note" type="string">Your ads refer
                         Free Minutes for new customers. What are Free
                         Minutes?
                         Thanks,Ed</pair>
                    </pair>
               </pair>
          </parameter>
```

```
        </function>
</connector>
```

In this example, the contact_id variable is set when the c_id is returned by the contact_create function. When the incident is created, the c_id of the newly created contact is passed using the variable $contact_id in the c_id pair.

Finding code numbers

You will frequently need to use code numbers in your XML to identify items such as products, categories, custom fields, and staff accounts. RightNow CRM provides two easy ways to look up the codes for these types of fields: mouseover functionality and the lookup_id_for_name function.

Using the mouseover function

You can use RightNow CRM mouseover functionality to look up many of the code numbers you need. Simply mouse over a profile, group, staff account, contact type, country, state, province, organization address type, service product, service category, incident disposition, incident status, answer status, answer access level, billable task, service level agreement, or custom field for each item accessed through the Management and Configuration Console.

Figure 1 shows the mouseover function for staff accounts. In this example, Dani Lion's account ID number (or code) is 4. This number is used to identify Dani when creating or updating records in RightNow CRM.

Path: >Common Configuration>Staff Management>Staff Accounts

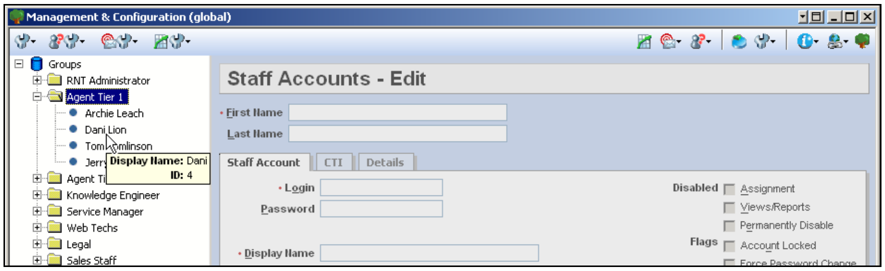
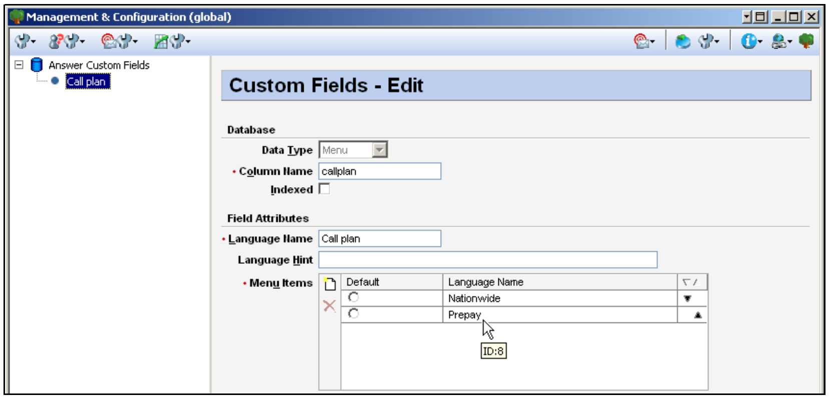


Figure 1: Mousing Over a Staff Account

Figure 2 shows the mouseover function for a custom field menu item. In this example, a menu item (Prepay) within the answer custom field "Call plan" is being referenced. The mouseover function shows that the menu item "Prepay" is associated with ID number (or code) 8, which is used to identify the "Prepay" menu item ID when creating or updating records in RightNow CRM.

Path: >Service Configuration>Custom Fields>Answer



Database

Data Type	Menu
Column Name	callplan
Indexed	☐

Field Attributes

Language Name	Call plan
Language Hint	

Menu Items

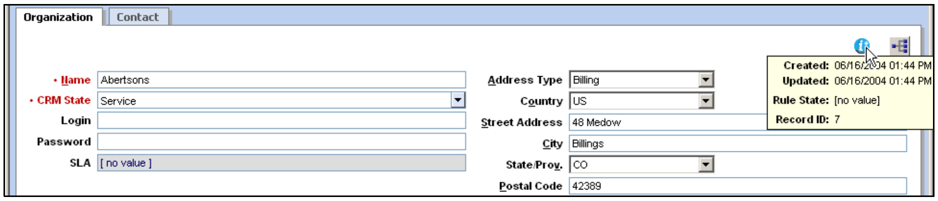
Default	Language Name
○	Nationwide
○	Prepay

ID:8

Figure 2: Mousing Over a Custom Field Menu Item

Figure 3 shows the mouseover function for an organization on the Organization tab on the Support Console. In this example, the organization's record ID is the code number. The mouseover function can also be used on the Contact tab on the Support Console, and the Organization and Contact tabs on the Sales Console.

Path: Quick Search >Organization Search> Search



Name	Abertsons
CRM State	Service
Login	
Password	
SLA	[no value]

Address Type	Billing
Country	US
Street Address	48 Medow
City	Billings
State/Prov.	CO
Postal Code	42389

Created: 06/16/2004 01:44 PM
Updated: 06/16/2004 01:44 PM
Rule State: [no value]
Record ID: 7

Figure 3: Mousing Over the Information Icon on the Organization Tab

Using the lookup_id_for_name function

In addition to using the mouseover functionality, you can also use an XML function, lookup_id_for_name, which will find the code number of an item and return the value by email or in a variable used later in your XML. This function can pass three parameters, lk_str, lk_tbl, and lk_fld. The lk_str parameter is used to pass the name of the string.

The lk_tbl parameter is used to pass the number of the table the code item belongs to. The lk_fld parameter is used to pass the name of a cf_id menu-item field you want to look up. The numbers of each table are listed in Table 9, along with the field looked up by the function.

Table 9: Table Numbers for lk_tbl Parameter

Table Name	Number	Lookup Field
incidents	1	ref_no
contacts	2	email
orgs	3	name
products (hier_menu)	13	name
categories (hier_menu)	14	name
menu_items	20	name
accounts	24	login

Example:

```
<connector ret_type="email" ret_email="staff@custhelp.com">
     <function name="lookup_id_for_name" id="prodid">
          <parameter name="lk_tbl" type="integer">13</parameter>
          <parameter name="lk_str" type="string">Product Name</parameter>
     </function>
     <function name="incident_update">
          <parameter name="args" type="pair">
               <pair name="prod_lvl1" type="integer">$prodid</pair>
          </parameter>
          <parameter name="i_id" type="integer">9</parameter>
     </function>
</connector>
```

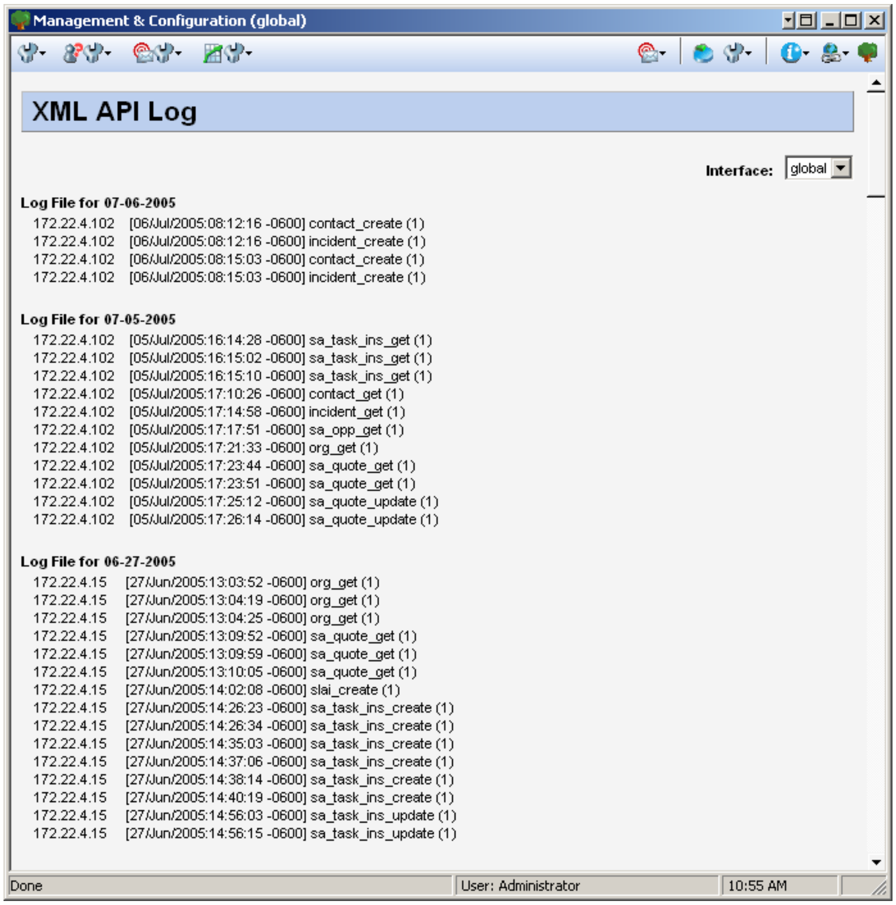
In this example, the function looks up the code number for the product, “Product Name,” and uses a variable, “prodid,” to use this code to update incident number 9. The product code will also be emailed to staff@custhelp.com.

Using the XML API log

The XML API log allows you to view a record of all XML functions passed to your Right-Now CRM site though the API. Each function that was performed through the XML API is listed, along with the IP address that passed the function, and the date and time it was performed. All functions are listed, regardless of whether an error occurred during the processing of the function.

The XML API log is accessed through the Management and Configuration console on the administration interface. You can use the log to track activity through the XML API and ensure security is maintained by monitoring the IP addresses that pass functions to your site.

Path: >Common Configuration>System Configuration>XML API Log



```
Log File for 07-06-2005
  172.22.4.102   [06/Jul/2005:08:12:16 -0600] contact_create (1)
  172.22.4.102   [06/Jul/2005:08:12:16 -0600] incident_create (1)
  172.22.4.102   [06/Jul/2005:08:15:03 -0600] contact_create (1)
  172.22.4.102   [06/Jul/2005:08:15:03 -0600] incident_create (1)

Log File for 07-05-2005
  172.22.4.102   [05/Jul/2005:16:14:28 -0600] sa_task_ins_get (1)
  172.22.4.102   [05/Jul/2005:16:15:02 -0600] sa_task_ins_get (1)
  172.22.4.102   [05/Jul/2005:16:15:10 -0600] sa_task_ins_get (1)
  172.22.4.102   [05/Jul/2005:17:10:26 -0600] contact_get (1)
  172.22.4.102   [05/Jul/2005:17:14:58 -0600] incident_get (1)
  172.22.4.102   [05/Jul/2005:17:17:51 -0600] sa_opp_get (1)
  172.22.4.102   [05/Jul/2005:17:21:33 -0600] org_get (1)
  172.22.4.102   [05/Jul/2005:17:23:44 -0600] sa_quote_get (1)
  172.22.4.102   [05/Jul/2005:17:23:51 -0600] sa_quote_get (1)
  172.22.4.102   [05/Jul/2005:17:25:12 -0600] sa_quote_update (1)
  172.22.4.102   [05/Jul/2005:17:26:14 -0600] sa_quote_update (1)

Log File for 06-27-2005
  172.22.4.15    [27/Jun/2005:13:03:52 -0600] org_get (1)
  172.22.4.15    [27/Jun/2005:13:04:19 -0600] org_get (1)
  172.22.4.15    [27/Jun/2005:13:04:25 -0600] org_get (1)
  172.22.4.15    [27/Jun/2005:13:09:52 -0600] sa_quote_get (1)
  172.22.4.15    [27/Jun/2005:13:09:59 -0600] sa_quote_get (1)
  172.22.4.15    [27/Jun/2005:13:10:05 -0600] sa_quote_get (1)
  172.22.4.15    [27/Jun/2005:14:02:08 -0600] slai_create (1)
  172.22.4.15    [27/Jun/2005:14:26:23 -0600] sa_task_ins_create (1)
  172.22.4.15    [27/Jun/2005:14:26:34 -0600] sa_task_ins_create (1)
  172.22.4.15    [27/Jun/2005:14:35:03 -0600] sa_task_ins_create (1)
  172.22.4.15    [27/Jun/2005:14:37:06 -0600] sa_task_ins_create (1)
  172.22.4.15    [27/Jun/2005:14:38:14 -0600] sa_task_ins_create (1)
  172.22.4.15    [27/Jun/2005:14:40:19 -0600] sa_task_ins_create (1)
  172.22.4.15    [27/Jun/2005:14:56:03 -0600] sa_task_ins_update (1)
  172.22.4.15    [27/Jun/2005:14:56:15 -0600] sa_task_ins_update (1)
```

Figure 4: XML API Log

4 Event Handlers

Through the RightNow CRM external event feature, you can define custom processes for managing your incidents, contacts, organizations, answers, and opportunities. For example, if you need to maintain your own incident database, you can create an event handler that automatically copies new incidents from RightNow CRM to your external database. These types of event handlers are ideal for building real-time interfaces between RightNow CRM and external help desks, call centers, data mining, or reporting systems.

The following types of external events are supported by RightNow CRM:

- **Insert events**—This type of event occurs whenever a customer or staff member creates a new incident, answer, contact, organization, or opportunity.
- **Update events**—This type of event occurs whenever a customer or a staff member updates an existing incident, answer, contact, organization, or opportunity.
- **Delete events**—This type of event occurs whenever a customer or a staff member deletes an existing incident, answer, contact, organization, or opportunity.

There are two ways to handle external events in RightNow CRM. You can specify the location of a script that directs the handling of an event (external events), or you can email the event data to a specified mailbox (application bridge). This chapter contains procedures for both of these methods.

Caution The files output by both external events and the application bridge are determined by template files within RightNow CRM. To implement external events, you must contact RightNow Technologies Professional Services to enable and customize these files. Failure to enable the template files could result in database errors. For more information, contact your RightNow account manager.

External events

The external event handlers can be enabled to run a specified script or program when an incident, answer, contact, organization, or opportunity is created, updated, or deleted. When an external event occurs, a data file (CSV) is created. The data is then handled under the direction of the script or program specified in your configuration settings. For example, you can create a script that will export specified incident data to an external Oracle database.

Caution The files output by external events are determined by template files within RightNow CRM. Before implementing external events, these files must be enabled and customized by RightNow Technologies Professional Services.

Enabling external events

Enabling external events requires configuring the insert, update, or delete handlers. The event handler configuration settings are located in RightNow CRM under RNT Common>External Events. These settings are described in Table 10.

Table 10: External Events Configuration Settings

Setting	Usage
EE_INC_DELETE_HANDLER	Specifies the full path name of a script or program to be used to externally process incident delete events. This is used to provide an interface for third-party call management systems or other third-party systems. If no handler is specified, no external action is taken. Default is blank.
EE_INC_INSERT_HANDLER	Specifies the full path name of a script or program to be used to externally process incident insert events. This is used to provide an interface for third-party call management systems or other third-party systems. If no handler is specified, no external action is taken. Default is blank.
EE_INC_UPDATE_HANDLER	Specifies the full path name of a script or program to be used to externally process incident update events. This is used to provide an interface for third-party call management systems or other third-party systems. If no handler is specified, no external action is taken. Default is blank.

Table 10: External Events Configuration Settings (Continued)

Setting	Usage
EE_CONTACT_DELETE_HANDLER	Specifies the full path name of a script or program to be used to externally process contact delete events. This is used to provide an interface for third-party call management systems or other third-party systems. If no handler is specified, no external action is taken. Default is blank.
EE_CONTACT_INSERT_HANDLER	Specifies the full path name of a script or program to be used to externally process contact insert events. This is used to provide an interface for third-party call management systems or other third-party systems. If no handler is specified, no external action is taken. Default is blank.
EE_CONTACT_UPDATE_HANDLER	Specifies the full path name of a script or program to be used to externally process contact update events. This is used to provide an interface for third-party call management systems or other third-party systems. If no handler is specified, no external action is taken. Default is blank.
EE_ANS_DELETE_HANDLER	Specifies the full path name of a script or program to be used to externally process answer delete events. This is used to provide an interface for third-party call management systems or other third-party systems. If no handler is specified, no external action is taken. Default is blank.
EE_ANS_INSERT_HANDLER	Specifies the full path name of a script or program to be used to externally process answer insert events. This is used to provide an interface for third-party call management systems or other third-party systems. If no handler is specified, no external action is taken. Default is blank.
EE_ANS_UPDATE_HANDLER	Specifies the full path name of a script or program to be used to externally process answer update events. This is used to provide an interface for third-party call management systems or other third-party systems. If no handler is specified, no external action is taken. Default is blank.
EE_ORG_DELETE_HANDLER	Specifies the full path name of a script or program to be used to externally process organization delete events. This is used to provide an interface for third-party call management systems or other third-party systems. If no handler is specified, no external action is taken. Default is blank.

Table 10: External Events Configuration Settings (Continued)

Setting	Usage
EE_ORG_INSERT_HANDLER	Specifies the full path name of a script or program to be used to externally process organization insert events. This is used to provide an interface for third-party call management systems or other third-party systems. If no handler is specified, no external action is taken. Default is blank.
EE_ORG_UPDATE_HANDLER	Specifies the full path name of a script or program to be used to externally process organization update events. This is used to provide an interface for third-party call management systems or other third-party systems. If no handler is specified, no external action is taken. Default is blank.
EE_OPP_DELETE_HANDLER	Specifies the full path name of a script or program used to externally process opportunity delete events. This is used to provide an interface for third party call management systems or other third party systems. If no handler is specified, no external action is taken. Default is blank.
EE_OPP_INSERT_HANDLER	Specifies the full path name of a script or program used to externally process opportunity insert events. This is used to provide an interface for third party call management systems or other third party systems. If no handler is specified, no external action is taken. Default is blank.
EE_OPP_UPDATE_HANDLER	Specifies the full path name of a script or program used to externally process opportunity update events. This is used to provide an interface for third party call management systems or other third party systems. If no handler is specified, no external action is taken. Default is blank.

Developing external events

When you activate an insert, update, or delete handler, the script or program you develop to handle the event becomes an extension of RightNow CRM. Event handlers can be written in Perl, C, Visual Basic, shell script, or any other programming language that can open and read a file and produce a standalone executable.

When developing event handlers, RightNow CRM will:

1 Create a CSV file. The file format will be the same as that produced by the *kexport* utility. The data field names and actual data provided will be determined according to the template file. For more information about *kexport*, refer to Chapter 13, "Utilities," in the *RightNow CRM 7.0 System Configuration Manual*.

2 Execute the appropriate script or program for handling the insert, update, or delete event. The names of the temporary files containing the incident, answer, contact, organization, or opportunity data are passed as the first two command line arguments.

Note Your event handler must reside either on your web server or in a network directory that your web server can access directly.

3 Wait for the event handler to terminate, and then continue with normal processing.

Your custom event handler will:

1 Read the first two command line arguments to get the names of the temporary files containing incident, answer, contact, organization, or opportunity data.

2 Open and parse the temporary files to retrieve incident, answer, contact, organization, or opportunity data.

3 Perform any custom processing.

4 Delete the temporary files.

5 Terminate and return control to RightNow CRM within an acceptable amount of time so as not to degrade overall system performance.

Application bridge

The application bridge in RightNow CRM allows you to email data to a mailbox when an incident, answer, contact, organization, or opportunity is created, updated, or deleted. When the event occurs, an email is sent immediately to the specified mailbox with the incident, answer, contact, organization, or opportunity data.

Enabling email integration requires configuring the insert, update, or delete handlers. The email integration configuration settings are located in RightNow CRM under RNT Common>External Events>Email Integration. Use these settings to specify the email address to which you want to send email event data.

The email integration configuration settings are described in Table 11.

Table 11: Email Integration Configuration Settings

Setting	Description
EI_INC_DELETE_ADDR	Specifies the email address to receive incident delete data. If no address is specified, no external action is taken. Default is blank.
EI_INC_INSERT_ADDR	Specifies the email address to receive incident insert data. If no address is specified, no external action is taken. Default is blank.
EI_INC_UPDATE_ADDR	Specifies the email address to receive incident update data. If no address is specified, no external action is taken. Default is blank.
EI_CONTACT_DELETE_ADDR	Specifies the email address to receive contact delete data. If no address is specified, no external action is taken. Default is blank.
EI_CONTACT_INSERT_ADDR	Specifies the email address to receive contact insert data. If no address is specified, no external action is taken. Default is blank.
EI_CONTACT_UPDATE_ADDR	Specifies the email address to receive contact update data. If no address is specified, no external action is taken. Default is blank.
EI_ANS_DELETE_ADDR	Specifies the email address to receive answer delete data. If no address is specified, no external action is taken. Default is blank.

Table 11: Email Integration Configuration Settings (Continued)

Setting	Description
EI_ANS_INSERT_ADDR	Specifies the email address to receive answer insert data. If no address is specified, no external action is taken. Default is blank.
EI_ANS_UPDATE_ADDR	Specifies the email address to receive answer update data. If no address is specified, no external action is taken. Default is blank.
EI_ORG_DELETE_ADDR	Specifies the email address to receive organization delete data. If no address is specified, no external action is taken. Default is blank.
EI_ORG_INSERT_ADDR	Specifies the email address to receive organization insert data. If no address is specified, no external action is taken. Default is blank.
EI_ORG_UPDATE_ADDR	Specifies the email address to receive organization update data. If no address is specified, no external action is taken. Default is blank.
EI_OPP_INSERT_ADDR	Specifies the email address to receive opportunity insert data. If no address is specified, no external action is taken. Default is blank.
EI_OPP_UPDATE_ADDR	Specifies the email address to receive opportunity update data. If no address is specified, no external action is taken. Default is blank.
EI_OPP_DELETE_ADDR	Specifies the email address to receive opportunity delete data. If no address is specified, no external action is taken. Default is blank.

Creating templates for the application bridge

When using the application bridge in RightNow CRM, you can create template files that specify the data sent by email following an event. You can create up to five template files and upload them to the *integration files* directory in File Manager. For more information about uploading files through the File Manager, refer to Chapter 7, “System Configuration,” in the *RightNow CRM 7.0 System Configuration Manual.*

To upload a file to the *integration files* directory in File Manager, the file must be in the following format:

- *incident.tmpl*—This template determines the data sent when an incident event (create, update, or delete) occurs.
- *ans.tmpl*—This template determines the data sent when an answer event (create, update, or delete) occurs.
- *contact.tmpl*—This template determines the data sent when a contact event (create, update, or delete) occurs.
- *org.tmpl*—This template determines the data sent when an organization event (create, update, or delete) occurs.
- *opp.tmpl*—This template determines the data sent when an opportunity event (create, update, or delete) occurs.

The template file will contain three components. The first line of the template specifies the reply-to address of the email. The second line of the template specifies the subject of the email. The remaining lines determine the content of the email. These lines can contain actual text, as well as variable information designated in pipes (|). Any text contained in pipes should be in the format table_name.column_name.

Important You can specify any field definition columns in the table related to the external event (*answers, contacts, incidents, orgs,* or *sa_opportunities*). You can also define output for any table directly related to the external event table. For example, you can require contact output in the *incident.tmpl* file because a contact should be directly related to each incident. For a list of the tables and columns in RightNow CRM, refer to Appendix D, "Database Schema Tables," on page 135.

The following is an example of an *incident.tmpl* file:

```
email@rightnow.com
Email Integration
Reference Number: |incidents.ref_no|
Subject: |incidents.subject|
Product: |incidents.prod_lvl1|
         |incidents.prod_lvl2|
```

In this example, the reply-to address of the email will be "email@rightnow.com" and the subject line of the email will be "Email Integration." The body of the email will look like the following:

```
Reference Number: 010620-000003
Subject: Incident Title
Product: Integration
```

5

Pass-Through Authentication

You can integrate RightNow Service with an external customer validation source to allow your customers to automatically log in to RightNow Service from an external web page. The external source supplies login parameters to RightNow Service by placing them in the URL of the Support Home page. In this way, customers will not have to provide customer login data twice if you are using an external customer validation source. The contact information will also be shared between the external source and RightNow Service, so contacts can be created and updated during the login to RightNow Service.

To perform this integration, customers must be redirected when attempting to access or log in to RightNow Service. When the login parameters are passed to RightNow Service, the customer will be logged in if the information passed is sufficient to identify an existing contact or create a new contact. An existing contact is identified by matching the email field and login field of the *contacts* table in the database. When an existing contact is found, the customer is logged in as that contact and is updated if any additional or new contact information is passed to RightNow Service.

If an existing contact is not found, a new contact is created from the data provided and the customer is logged in to RightNow Service as the new contact. If the contact information passed does not contain all required fields to create a new contact, RightNow Service can be configured to redirect the customer to an alternate URL.

Important When using pass-through authentication, the configuration setting EGW_AUTO_CUST_CREATE should be set to No to prevent contact records from being created through email before they are created by a pass-through authentication event. This will help eliminate login issues caused by mismatched user names and passwords.

Note If you set EGW_AUTO_CUST_CREATE to No, you should also modify the message base NOT_REG_EMAIL_MSG to direct new end-users to your portal sight to register and create an account.

Important Once logged in to RightNow Service, if the customer selects the My Stuff tab, the profile button is removed from this page ensuring the customer cannot update their information via the end-user interface.

Although contacts can be created and updated through the pass-through authentication integration, deletion of contacts must be handled by manually deleting the contact from RightNow Service though the Support Console or another integration method.

Note Contact your RightNow account manager for assistance in customizing your pass-through authentication beyond the procedures detailed in this chapter (for example, securing pass-through authentication strings beyond Base 64 encryption standards).

Refer to Figure 5 for assistance in designing your login integration. This figure can help you determine the process used by RightNow Service when pass-through authentication is used.

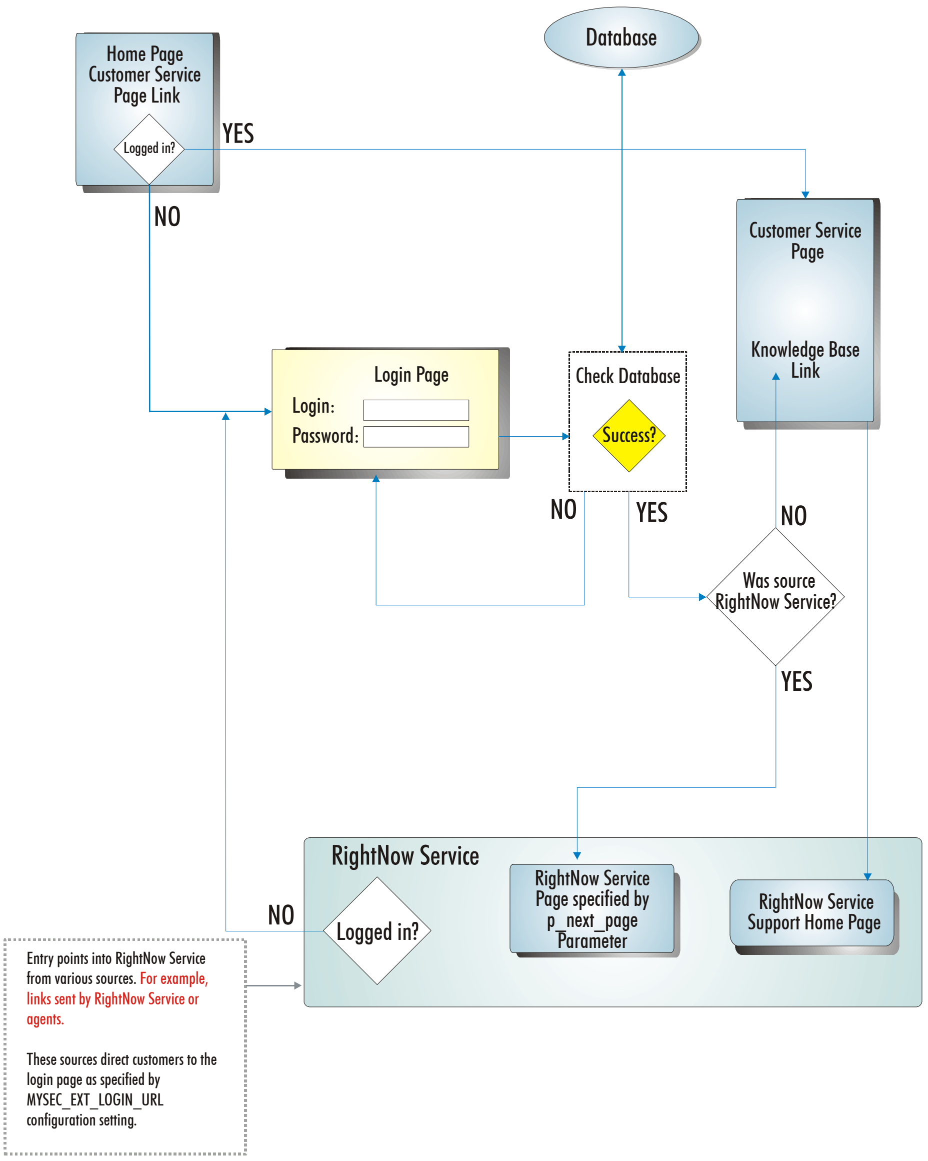

Home Page Customer Service Page Link
Logged in?
YES
NO
Database
Customer Service Page
Knowledge Base Link
Login Page
Login:
Password:
Check Database
Success?
NO
YES
NO
Was source RightNow Service?
YES
RightNow Service
NO
Logged in?
RightNow Service Page specified by p_next_page Parameter
RightNow Service Support Home Page
Entry points into RightNow Service from various sources. For example, links sent by RightNow Service or agents.
These sources direct customers to the login page as specified by MYSEC_EXT_LOGIN_URL configuration setting.

Figure 5: Pass-Through Authentication Flow Chart

Configuring RightNow Service

Before you can perform a pass-through authentication integration with an external source, you must configure RightNow Service to prevent customers from accessing specific options without a proper login. Either of the following methods can be used to configure RightNow Service to best suit the needs of your organization.

- **Require a login to RightNow Service**—This option configures RightNow Service so customers cannot access any end-user page without first logging in through your external validation source. Refer to "Requiring a login to RightNow Service" on page 92.
- **Disable contact account creation**—This options allows customers to access the RightNow Service knowledge base without logging in. However, customers must log in to submit a question through the Ask a Question page or access the My Stuff pages. Refer to "Disabling contact account creation" on page 93.

With either method, you must redirect the login to the URL of your external validation source. For more information, refer to "Redirecting the RightNow Service login" on page 93.

Requiring a login to RightNow Service

To integrate RightNow Service with an external customer validation source, RightNow Service can be configured to require a login to the end-user interface (excluding the Site Feedback page). This ensures that contact information is passed directly to the login page and prevents customers from accessing their account information through the end-user pages. However, customers will also be required to log in when clicking the link in an incident email to respond or update their incident.

To configure RightNow Service to require a login:

1 Click >Common Configuration>System Configuration>Settings.

2 Select RNT User Interface.

3 Click End-User Interface>Support Home Page Display>SHP_PASSWD_REQD, and click Yes for the value (No is the default).

4 Click Update followed by Commit and Exit to save your changes and return to the General Configuration Menu.

Disabling contact account creation

RightNow Service can also be configured to redirect contacts if they attempt to submit a question through the Ask a Question page or access the My Stuff page. In this way, your customers are free to search your knowledge base without being required to log in through your external validation source.

To configure RightNow Service to require a login:

1. Click >Common Configuration>System Configuration>Settings.
2. Select RNT User Interface.
3. Click My Stuff>Security>MYSEC_AUTO_CUST_CREATE and click No for the value (Yes is the default).
4. Click **Update** followed by **Commit and Exit** to save your changes and return to the General Configuration Menu.

Redirecting the RightNow Service login

A configuration setting must also be enabled when using the integration to specify the URL to which a customer is redirected if attempting to log in to RightNow Service, or if the external login information supplied to RightNow Service is not adequate to create a new account or use an existing account. When a URL value is specified for this configuration setting, the passed login parameters must provide data for the minimum required fields needed to log in to RightNow Service (p_userid, p_passwd) or create a new contact in RightNow Service (p_userid, p_passwd, p_email). Even if the configuration setting CT_EMAIL_REQD is disabled, the specified fields are still required. If the required fields are not passed, the customer is redirected to the specified URL.

Required If additional required contact custom fields have been created, these will also need to passed to create a new account.

You can create a new site at this URL to either inform the customer that their access is denied or create a form to gather additional required information and re-pass the parameters to RightNow Service.

Note RightNow Service will automatically append your customer's session ID information to the URL when the customer is redirected through the end-user pages. The page specified must be configured to accept the session ID.

To configure RightNow Service to redirect the login:

1. Click >Common Configuration>System Configuration>Settings.
2. Select RNT User Interface.
3. Click My Stuff>Security>MYSEC_EXT_LOGIN_URL.
4. Type the desired URL in the Value text box and click Update.
5. Click Commit and Exit to save your changes and return to the General Configuration Menu.

Note URLs sent to contacts via email (for example, a link to update the incident) will use the URL specified in the MYSEC_EXT_LOGIN_URL configuration setting.

If you are passing a non-blank password via p_passwd in a pass-through authentication event and DE_CUST_PASSWD_ENABLED is disabled, the pass-through authentication event will fail. It is recommended that you enable DE_CUST_PASSWD_ENABLED when using pass-through authentication and use CT_PASSWD_DISP to control the look and feel of contact passwords on the administration side of RightNow CRM. CT_PASSWD_DISP does not affect pass-through authentication.

Implementing a customer login script

To develop a login parameters integration, you will need to embed code within your login script to format a URL that will pass data from your external validation source to RightNow Service. The embedded code can be written in any scripting language, including PHP, JSP, or ASP. The login parameters from the external validation source must be encoded using Base 64 encryption and placed in the RightNow Service URL from the desired page. In addition to using the Base 64 function, certain characters must also be replaced in the URL, as shown in "Example:" on page 98 (+ becomes _, / becomes -, and = becomes *).

Note You must use a login script for every link from your web site to RightNow Service. If contacts exit the RightNow Service end-user pages and re-enter later in their session, they will not be automatically logged in. Therefore, we recommend that all links to the end-user interface contain pass-through data.

The following format should be used:

UNIX:

```
http://<your_domain>/cgi-bin/<your_interface>.cfg/php/enduser/
entry.php?p_li=<encoded login parameters>
```

Windows:

```
http://<your_domain>/scripts/<your_interface>.cfg/php.exe/enduser/
entry.php?p_li=<encoded login parameters>
```

Note You can replace entry.php with any end-user page in RightNow Service (for example, std_alp.php), or use the p_next_page parameter to return the customer to their original RightNow Service page. Refer to "Example:" on page 98.

The parameters to be passed to RightNow Service are detailed in Table 12.

Table 12: Parameter Descriptions

Parameter	Description
p_userid	This parameter represents the login field in the *contacts* table of the RightNow database. This field is required to log in and create a new contact, and cannot be updated via pass-through authentication.

Table 12: Parameter Descriptions (Continued)

Parameter	Description
p_passwd	This parameter represents the password field in the *contacts* table of the RightNow database (limited to 20 characters). This field is required to log in and create a new contact, or log in as an existing contact, and cannot be updated via pass-through authentication. The value can be NULL. **Note:** We recommend that the password specified in the *contacts* table be different than that stored in your external database. This is because the customer's RightNow Service password cannot be updated later by the external system, since the password is used as a verification field by RightNow Service. Therefore, to prevent customers who change their password in your external system from being locked out of the RightNow Service end-user pages, you should create a different password when the contact is created, and use this password consistently to log in the customer to RightNow. One way to accomplish this is to use a constant value for all contact passwords and use the value each time a customer logs in. You could also encrypt the contact's user id and use the encryption as the contact's password. Each time the customer's login parameters are passed to RightNow Service, you can use your encryption script to pass the valid password.
p_email	This parameter represents the email field in the *contacts* table in the RightNow database. This field is required to log in and create a new contact. **Note:** The value of this field must be unique.
p_first_name	This parameter represents the first_name field in the *contacts* table in the RightNow database.
p_last_name	This parameter represents the last_name field in the *contacts* table in the RightNow database.
p_email_alt1	This parameter represents the email_alt1 field in the *contacts* table in the RightNow database.
p_email_alt2	This parameter represents the email_alt2 field in the *contacts* table in the RightNow database.
p_street	This parameter represents the street field in the *contacts* table in the RightNow database.
p_city	This parameter represents the city field in the *contacts* table in the RightNow database.

Table 12: Parameter Descriptions (Continued)

Parameter	Description
p_postal_code	This parameter represents the postal_code field in the *contacts* table in the RightNow database. This field may not contain special characters (for example, 59715-1111 should be passed as 597151111).
p_country_id	This parameter represents the country_id field in the *contacts* table in the RightNow database. This field should be passed as a country's ID number. To find the value of menu items, refer to "Finding code numbers" on page 75.
p_prov_id	This parameter represents the prov_id field in the *contacts* table in the RightNow database. This field should be passed as a state or province's ID number. To find the value of menu items, refer to "Finding code numbers" on page 75.
p_ph_office	This parameter represents the ph_office field in the *contacts* table in the RightNow database. This field may not contain special characters (for example, (406)555-5555 should be passed as 4065555555).
p_ph_mobile	This parameter represents the ph_mobile field in the *contacts* table in the RightNow database. This field may not contain special characters (for example, (406)555-5555 should be passed as 4065555555).
p_ph_fax	This parameter represents the ph_fax field in the *contacts* table in the RightNow database. This field may not contain special characters (for example, (406)555-5555 should be passed as 4065555555).
p_ph_asst	This parameter represents the ph_asst field in the *contacts* table in the RightNow database. This field may not contain special characters (for example, (406)555-5555 should be passed as 4065555555).
p_ph_home	This parameter represents the ph_home field in the *contacts* table in the RightNow database. This field may not contain special characters (for example, (406)555-5555 should be passed as 4065555555).
p_ccf_*	The parameter p_ccf_* represents a contact custom field in RightNow CRM. The * should be replaced with the number of the cf_id for the contact custom field. If this is a menu custom field, the numbers (not the actual text) for each menu item must be specified as the value in the integration login code. To find the value of menu items, refer to "Finding code numbers" on page 75.

Table 12: Parameter Descriptions (Continued)

Parameter	Description
p_li_passwd	This parameter represents the string specified in the MYSEC_LI_PASSWD configuration setting. This parameter is required if the MYSEC_LI_PASSWD configuration setting contains a value.
p_org_id	This parameter represents an organization ID to associate with a contact. To find the value of menu items, refer to "Finding code numbers" on page 75. **Note:** You must manually assign any service level agreements (SLA) that you want to associate with the organization, including those controlling privileged access. You can do this through RightNow CRM's administration interface.

p_li_expiry	This parameter represents the time the login session will last before expiring. When the session expires, the contact will be required to resubmit their login on the page specified by the MYSEC_EXT_LOGIN_URL configuration setting. The following format should be used:
	p_li_expiry=time()+<seconds>
	where <seconds> is the time, in seconds, before the session expires. For example, if you want the login session to last one hour, pass the following statement:
	p_li_expiry=time()+3600

The following is an example of how to generate a form to pass login parameters to RightNow Service using PHP code. You can retain all query_string parameters and append key-value pair parameters per the following example.

Caution The following example may be improperly formatted if you attempt to cut and paste directly from the following text. To ensure proper functioning, review the script format before implementing it on your support site.

Example:

```
<?
// li.php
//
// ***** THIS IS JUST AN EXAMPLE AND NOT INTENDED FOR PRODUCTION USE *****
//Use this script to see an illustrated example of how login integration
//is supposed to work. This script will generate a form that requests a
//login/password and other optional information. It submits this data
```

```
//back to itself (with $li_reentry set), sets up the appropriate
//parameters (important ones passed in from RNW) and redirects
//back to RNW
// ----------------------------------------------------------
// Site specific variables

$script_name = 'li.php';
$domain = '<your_domain>';
$script_dir = '<cgi-bin or scripts>';
$interface = '<your_interface>';
$mysec_li_passwd = '<li_password>';
$php_bin = '<php or php.exe>';
// ----------------------------------------------------------
// Function definitions

function urlsafe_encode(&$str)
{
     return(strtr(base64_encode($str),
          array('+' => '_', '/' => '-', '=' => '*')));
}

function urlsafe_decode(&$str)
{
     return(base64_decode(strtr($str,
          array('_' => '+', '-' => '/', '*' => '='))));
}
// ----------------------------------------------------------
// Process the form & redirect

if ($li_reentry) {
     $li_data = array(
          'p_userid'     => $li_userid,
          'p_passwd'     => $li_passwd,
          'p_email'      => $li_email,
          'p_first_name' => $li_first_name,
          'p_last_name'  => $li_last_name,
     // sample text contact custom field (custom_fields.cf_id== 1)
          'p_ccf_1'      => $li_ccf_1,
     // sample menu contact custom field (custom_fields.cf_id == 3)
          'p_ccf_3'      => intval($li_ccf_3),
          // p_li_passwd must match the MYSEC_LI_PASSWD config setting
               'p_li_passwd'  => $mysec_li_passwd
          );

          // set up the $p_li variable
          while (list($key, $val) = each($li_data))
               $p_li .= sprintf("%s%s=%s", $p_li ? '&' : '', $key,
               $val);
          $p_li = urlsafe_encode($p_li);

          // retain all the important query_string parameters passed in from
          //RNW (excluding the special cases and the li_* form parameters)
          while (list($key, $val) = each($HTTP_GET_VARS)) {
               if (($key != 'p_next_page') &&
                    ($key != 'p_li') &&
                    (substr($key, 0, 3) != 'li_'))
                    $parms .= sprintf("&%s=%s", $key,
                    urlencode($val));
          }

          // default next page to support home
          if (!isset($p_next_page))
               $p_next_page = "home.php";
          // redirect back to RNW
          header("Location: http://$domain/$script_dir/       \
          $interface.cfg/$php_bin/enduser/$p_next_page?p_li   \
          =$p_li$parms");
          exit;
     }

     // -------------------------------------------------------------
     // Display the form
     ?>
     <html>
     <body>

     <h2> Login Integration </h2>

     <form action="<? print($script_name) ?>">
     <input type="hidden" name="li_reentry" value="1">
```

```
<?
// retain all the important query_string parameters passed in from RNW
while (list($key, $val) = each($HTTP_GET_VARS)) {
     print("<input type=\"hidden\" name=\"$key\"     \
     value=\"$val\">\n");
}
?>
Login: <input type="text" name="li_userid"><br />
Password: <input type="password" name="li_passwd"><br />
Email: <input type="text" name="li_email"><br />
First Name: <input type="text" name="li_first_name"><br />
Last Name: <input type="text" name="li_last_name"><br />
Contact Custom 1: <input type="text" name="li_ccf_1"><br />
Contact Custom 3: <input type="text" name="li_ccf_3"><br />
<input type="submit">
</form>

</body>
</html>
```

Note To implement this script, you must replace certain variables to correctly format your URL. Replace <your_domain> with the domain name used by RightNow Service, <your_interface> with your interface name, and <li_password> with the string specified in MYSEC_LI_PASSWD. In addition, specify "cgi-bin" and "php" for UNIX or "scripts" and "php.exe" for Windows.

Appendix A Pair Names

This appendix describes the pairs available to be used in the accounts, answers, contacts, custom fields, hierarchical menus, incidents, meta-answers, opportunities, organization, quotes, search, and tasks API.

Account API

The pairs described in the following table are available to use in account functions.

Table 13: Account Pairs

Name	Use	Type
acd_group	The automatic call distribution group associated with a staff account.	integer
acd_passwd	The automatic call distribution password associated with a staff account.	string
alt_first_name	The alternate first name of a staff account.	string
alt_last_name	The alternate last name of a staff account.	string
attr	A bitmap that determines the attribute statuses of the account. • 0—Fully enabled • 1—Assignment to the staff member is disabled • 2—Views and reports are disabled • 4—Account locked • 8—Force password change • 32—Permanently disabled	integer
country_id	The default country associated with a staff account.	integer

Table 13: Account Pairs (Continued)

Name	Use	Type
custom_field	A custom field associated with a staff account.	pair
dca_enabled	The Sales Console disconnected access flag associated with a staff account.	integer
def_currency	The default currency associated with a staff account.	integer
display_name	The display name associated with a staff account.	string
eas_id	The Avaya Expert Agent Selection (EAS) flag associated with a staff account.	integer
email_address	The email address associated with a staff account.	string
email_notif	The email notification flag associated with a staff account.	integer
first_name	The first name associated with a staff account.	string
group_id	The group ID associated with a staff account.	integer
last_name	The last name of a staff account.	string
login	The login associated with a staff account.	string
old_terr	The old territory associated with a staff account.	integer
password	The password associated with a staff account.	string
password_exp	The password expiration date of a staff account.	time
phone	The phone number associated with a staff account.	string

Table 13: Account Pairs (Continued)

Name	Use	Type
profile_id	The profile ID associated with a staff account.	integer
sa_def_cmode	The Sales Console mode the staff member starts in.	integer
seq	The sequence listing within a group folder that is associated with a staff account.	integer
signature	The signature associated with a staff account.	string
softphone	The Softphone flag associated with a staff account.	integer
sp_dial	The Softphone speed dial associated with a staff account.	string
start_console	The start console associated with a staff account.	integer
terr_id	The territory ID associated with a staff account.	integer
upd_opt	The flag that updates opportunities when changing the territory of a staff account.	integer

Answer API

The pairs described in the following table are available to use in answer functions.

Table 14: Answer Pairs

Name	Use	Type
access_mask	The answer access of the answer. The access level determines which end-users can view the answer.	string
assgn_acct_id	The ID number of the staff member assigned to the answer.	integer

Table 14: Answer Pairs (Continued)

Name	Use	Type
assgn_group_id	The ID number of the staff group assigned to the answer.	integer
custom_field	A custom field associated with the answer.	pair
description	The description of the answer.	string
expires	The date the answer expires and is set to review answer status.	time
keywords	The keywords of the answer.	string
lang_id	The ID number of the answer's language.	integer
last_access	The date and time the answer was last accessed.	time
last_edited_by	The ID number of the staff member who last edited the answer.	integer
last_notify	The date and time a notification was last sent for the answer.	time
next_notify	The date a notification will be sent for the answer.	time
notes	The notes field of the answer.	string
prev_access_id	The ID number of the answer access the answer was previously assigned to.	integer
prev_assgn_acct_id	The ID number of the staff member the answer was previously assigned to.	integer
prev_assgn_group_id	The ID number of the staff group the answer was previously assigned to.	integer
prev_status_id	The ID number of the status the answer was previously assigned to.	integer
publish_on	The date the answer will be published on.	time
rule_state	The rule state the answer is currently in.	integer

Table 14: Answer Pairs (Continued)

Name	Use	Type
solution	The solution of the answer.	string
solved_count	The relevancy ranking of this answer.	integer
source	The source of the answer.	integer
static_solved	The fixed relevancy ranking of this answer (100 is fix at top, 50 is fix at middle, 0 is fix at bottom).	integer
status_id	The status of the answer.	integer
status_type	The status type the answer is assigned to.	integer
summary	The title of the answer.	string

Campaign API

The pairs described in the following table are available to use in campaign functions.

Table 15: Campaign Pairs

Name	Use	Type
c_id	The ID number of the contact associated with the incident.	integer
campaign_id	The code number of the campaign.	integer
entry_point	The Shortcut ID field that is entered in the dialog for an Entry Point node in a campaign.	string

Contact API

The pairs described in the following table are available to use in contact functions.

Table 16: Contact Pairs

Name	Use	Type
cat_lvl<1-6>	The pair data of the default category for the contact's searching.	integer
city	The name of the city in the contact's address information.	string
country_id	The ID number of the country in the contact's address information.	integer
css_state	The Service state flag associated with the contact.	integer
ctype_id	The ID number of the contact type.	integer
custom_field	A custom field associated with the contact.	pair
disabled	The disabled status of the contact (1=disabled, 0=enabled).	integer
email	The primary email address of the contact.	string
email_alt1	The first alternate email address of the contact.	string
email_alt2	The second alternate email address of the contact.	string
first_name	The first name of the contact.	string
last_name	The last name of the contact.	string
lines_per_page	The default number of lines per page shown for a contact.	integer
login	The contact login name.	string
ma_list_ids	The marketing list IDs associated with the contact.	string

Table 16: Contact Pairs (Continued)

Name	Use	Type
ma_mail_type	The marketing mail type associated with the contact.	integer
ma_opt_in	The marketing opt-in flag associated with the contact.	integer
ma_org_name	The marketing organization name associated with the contact.	string
ma_state	The Marketing state flag associated with the contact.	integer
org_id	The ID number of the organization associated with the contact.	integer
password	The contact’s password.	string
ph_asst	The phone number of the contact’s assistant.	string
ph_fax	The contact’s fax number.	string
ph_home	The contact’s home phone number.	string
ph_mobile	The contact’s mobile phone number.	string
ph_office	The contact’s office phone number.	string
postal_code	The postal or zip code in the contact’s address.	string
prod_lvl<1-6>	The pair data of the default product for the contact’s searching.	integer
prov_id	The ID number of the province or state in the contact’s address information.	integer
rule_state	The rule state the contact is currently in.	integer
sa_state	The Sales state flag associated with the contact.	integer
search_text	The default search text for the contact’s searching.	string

Table 16: Contact Pairs (Continued)

Name	Use	Type
search_type	The code of the default search type for the contact's searching.	integer
sessionid	The session ID for the contact.	string
source	The creation source of the contact. (Refer to Appendix B, "Source Codes," on page 127.)	integer
street	The contact's street address.	string
thread	The threads (notes) associated with a contact.	string
title	The contact's title.	string
updated_by	The staff member the contact was last updated by.	integer

Custom Field API

The pairs described in the following table are available to use in custom field functions.

Table 17: Custom Field Pairs

Name	Use	Type
cf_id	The code number of a custom field.	integer
cf_item	A custom field item pair array.	pair
custom_field	A custom field pair array.	pair
value	The value of the custom field.	string

Hierarchical menu API

The pairs described in the following table are available to use in hieararchical menu functions.

Table 18: Hierarchical Menu Pairs

Name	Use	Type
admin	The visibility of the hierarchical menu item on the administration interface.	integer
desc	The description of the hierarchical menu item.	string
enduser	The visibility of the hierarchical menu item on the end-user interface.	integer
id	The ID number of the hierarchical menu item.	integer
label	The name of the hierarchical menu item.	string
lvl	The level of the hierarchical menu item which can range in value from 0-5.	integer
parent_id	The ID of the higher-level hierarchical menu item that the lower-level hierarchical menu item is associated with.	integer
seq	The position of the hierarchical menu item within the list of hierarchical menu items.	integer
tbl	The table code of the hierarchical menu item. Refer to Table 4 on page 39.	integer
vis	The visibility of the hierarchical menu item.Each set of visibility settings is contained in a vis_item pair.	pair
vis_item	The visibility settings for the hierarchical menu item. Each set of visibility settings includes the admin, enduser, and intf_id pairs.	pair

Incident API

The pairs described in the following table are available to use in incident functions.

Table 19: Incident Pairs

Name	Use	Type
assgn_acct_id	The ID number of the staff member the incident is assigned to.	integer
assgn_group_id	The ID number of the staff group the incident is assigned to.	integer
c_id	The ID number of the contact associated with the incident.	integer
cat_lvl<1-6>	The category assigned to the incident.	integer
created_by	The ID number of the incident creator.	integer
custom_field	A custom field associated with the incident.	pair
disp_lvl<1-6>	The disposition assigned to the incident.	integer
dormant	The dormant status of an incident (0 is not dormant, 1 is dormant).	integer
ei_cust	The emotive index of the contact associated with the incident.	integer
ei_staff	The emotive index of the staff assigned to the incident.	integer
entry_type	The type of incident thread (1 is note, 2 is staff entry, 3 is contact entry, and 4 is customer proxy).	integer
escldate	The date and time the incident was escalated.	time
escllevel	The level that the incident has been escalated to through the rules engine.	integer
initial_soln	The date and time the incident was responded to ending with a status change to a type other than unresolved.	time

Table 19: Incident Pairs (Continued)

Name	Use	Type
interface_id	The ID number of the interface associated with the incident.	integer
lang_id	The ID number of the language associated with the incident.	integer
last_resp	The date and time the incident was last responded to.	time
ma_mailing_id	The Marketing outbound email ID associated with the incident.	integer
mail_hdr	The mail header associated with the incident.	string
mailbox_id	The ID number of the mailbox the incident was created from.	integer
org_id	The ID number of the organization.	integer
prev_assgn_acct_id	The ID number of the staff member the incident was previously assigned to.	integer
prev_assgn_group_id	The ID number of the staff group the incident was previously assigned to.	integer
prev_queue_id	The ID number of the queue the incident was previously assigned to.	integer
prev_slai_id	The ID number of the SLA instance the incident was previously assigned to.	integer
prev_status_id	The ID number of the status the incident was previously assigned to.	integer
prev_status_type	The ID number of the status type the incident was previously assigned to.	integer
prod_lvl<1-6>	The product assigned to the incident.	integer
queue_id	The ID number of the queue the incident is assigned to.	integer
ref_no	The reference number of the incident.	string

Table 19: Incident Pairs (Continued)

Name	Use	Type
rel_due	The relative due date to be met to meet the SLA. If SLAs have not been implemented, this would apply to the default response requirements.	time
resp_sav	An uncommitted (not sent) response thread.	string
rnl_queue_id	The ID number of the RightNow Live queue the incident is assigned to.	integer
rule_state	The rule state the incident is currently in.	integer
send_resp	A committed and sent response thread.	integer
sessionid	The ID string of the end-user session the incident was created from.	string
sla_resp_delta	The number of minutes it took to respond to the incident past the SLA's response requirement.	integer
sla_rsln_delta	The number of minutes it took to resolve the incident past the SLA's resolution requirement.	integer
slai_id	The ID number of the SLA instance the incident is assigned to.	integer
source	The creation source of the incident. (Refer to Appendix B, "Source Codes," on page 127.)	integer
status_id	The status of the incident.	integer
status_type	The status type of the incident.	integer
subject	The title of the incident.	string
thread	The incident threads (response, note, customer).	pair
thread_entry	A part of the incident thread definition.	pair

Table 19: Incident Pairs (Continued)

Name	Use	Type
updated_by	The ID number of the staff member updating the incident.	integer
updated_by_c_id	The ID number of the contact updating the incident.	integer

Meta-answer API

The pairs described in the following table are available to use in meta-answer functions.

Table 20: Meta-Answer Pairs

Name	Use	Type
categories	The categories associated with the meta-answer.	pair
hier_item	The product or category.	pair
id1	The first-level product or category.	integer
id2	The second-level product or category.	integer
id3	The third-level product or category.	integer
id4	The fourth-level product or category.	integer
id5	The fifth-level product or category.	integer
id6	The sixth-level product or category.	integer
last_edited_by	The ID of the staff member who last edited the meta-answer.	integer
notes	The notes field of the meta-answer.	string
orig_ref_no	The original reference number of an incident that has been converted to an answer.	string
products	The products associated with the meta-answer.	pair
source	The source of the meta-answer.	integer

Table 20: Meta-Answer Pairs (Continued)

Name	Use	Type
summary	The title of the meta-answer.	string

Opportunity API

The pairs described in the following table are available to use in opportunity functions.

Table 21: Opportunity Pairs

Name	Use	Type
acct_lvl<1-12>_id	The ID number of the manager of the sales representative assigned to the opportunity.	integer
assgn_acct_id	The ID number of the sales representative assigned to the opportunity.	integer
assgn_group_id	The ID number of the group the sales representative assigned to the opportunity belongs to.	integer
c_id	The ID number of the contact associated with the opportunity.	integer
closed	The date and time the opportunity was closed.	time
closed_value	The closed value of the opportunity.	integer
closed_value_curr_id	The ID number of the currency of the closed value.	integer
closed_value_rate_id	The ID number of the exchange rate of the closed value.	integer
cr_id	The ID number of the contact role of a contact associated with the opportunity.	integer
created_by	The ID number of the staff member who created the opportunity.	integer
custom_field	A ID number of the custom field associated with the opportunity.	pair

Table 21: Opportunity Pairs (Continued)

Name	Use	Type
dormant	The dormant status of the opportunity (1=dormant, 0=not dormant).	integer
escldate	The date and time the opportunity was escalated.	time
escllevel	The level the opportunity was escalated to.	integer
forecast_close	The date the opportunity is forecasted to close.	time
initial_contact	The date the sales representative initially made contact with the organization.	time
interface_id	The ID number of the interface the opportunity is associated with.	integer
mgr_commit	The committed status of the manager-forecasted value (1=committed, 0=not committed).	integer
mgr_value	The manager-forecasted value of the opportunity.	integer
mgr_value_curr_id	The ID number of the currency of the manager-forecasted value.	integer
mgr_value_rate_id	The ID number of the exchange rate of the manager-forecasted value.	integer
name	The name of the opportunity.	string
oc_item0 through oc_item<x>	A contact associated with an opportunity. Contacts are defined with c_id, cr_id, and oc_primary pairs.	pair
oc_primary	Defines which contact is the primary contact associated with the opportunity. One contact must be specified as the primary contact. A value of 1 identifies the primary contact.	integer

Table 21: Opportunity Pairs (Continued)

Name	Use	Type
opp2contact	The contacts associated with an opportunity. Individual contacts are defined with the oc_item pair.	pair
org_id	The ID number of the organization associated with the opportunity.	integer
prev_acct_lvl<1-12>_id	The level of the sales representative who was previously assigned to the opportunity.	integer
prev_assgn_acct_id	The sales representative previously assigned to the opportunity.	integer
prev_assgn_group_id	The group ID of the sales representative previously assigned to the opportunity.	integer
prev_escllevel	The previous escalation level.	integer
prev_stage_id	The previous stage of the opportunity.	integer
prev_status_id	The previous status of the opportunity.	integer
prev_status_type	The previous status type of the opportunity.	integer
recall	The recall date associated with an opportunity.	time
rep_commit	The committed status of the sales representative-forecasted value (1=committed, 0=not committed).	integer
rep_value	The sales representative-forecasted value of the opportunity.	integer
rep_value_curr_id	The ID number of the currency of the sales representative-forecasted value.	integer
rep_value_rate_id	The ID number of the exchange rate of the sales representative-forecasted value.	integer
rule_state	The rule state the opportunity is associated with.	integer

Table 21: Opportunity Pairs (Continued)

Name	Use	Type
source	The creation source of the opportunity. (Refer to Appendix B, “Source Codes,” on page 127.)	integer
stage_id	The stage the opportunity is in.	integer
status_id	The status of the opportunity.	integer
status_type	The status type of the opportunity.	integer
strategy_id	The ID number of the opportunity’s strategy.	integer
summary	The summary line of the opportunity.	string
terr_id	The ID number of the territory associated with the opportunity.	integer
terr_lvl<1-12>_id	The hierarchical territories associated with the opportunity.	integer
thread	The notes threads (note, phone, email) associated with an opportunity.	pair
update	The date and time the opportunity was last updated.	time
updated_by	The ID number of the staff member who last updated the opportunity.	integer

Organization API

The pairs described in the following table are available to use in organization functions.

Table 22: Organization Pairs

Name	Use	Type
css_state	The Service state flag associated with the organization.	integer

Table 22: Organization Pairs (Continued)

Name	Use	Type
custom_field	A custom field associated with the organization.	pair
login	The organization login name.	string
ma_state	The Marketing state flag associated with the organization.	integer
name	The name of the organization.	string
oaddr	The addresses of the organization. Each address is contained in an oaddr_item pair.	pair
oaddr_item	An organization address.	pair
password	The password of the organization.	string
rule_state	The rule state the organization is currently in.	integer
sa_state	The Sales state flag associated with the organization.	integer
salesperson	The ID number of the sales representative who is associated with the organization.	integer
source	The creation source of the organization. (Refer to Appendix B, “Source Codes,” on page 127.)	integer
thread	The threads (notes) associated with an organization.	string
updated_by	The staff member who last updated the organization.	integer

Organization address item pairs

The pairs described in the following table are available to use in organization address items.

Table 23: Organization Address Item Pairs

Name	Use	Type
city	The city associated with the address.	string
country_id	The ID number of the country associated with the address.	integer
oat_id	The type of address (billing=1, ship-ping=2).	integer
postal_code	The postal or zip code associated with address.	string
prov_id	The ID number of the state or province associated with the address.	integer
street	The street address.	string

Quote API

The pairs described in the following table are available to use in quote functions.

Table 24: Quote Pairs

Name	Use	Type
created_by	The ID number of the staff member who created the quote.	integer
custom_field	A ID number of the custom field associ-ated with the quote.	pair
discount	The discount applied to the quote.	integer
forecast	The forecast status of the quote (1=Fore-cast check box is selected, 0=Forecast check box is cleared).	integer
name	The name of the quote.	string

Table 24: Quote Pairs (Continued)

Name	Use	Type
notes	The notes associated with the quote.	string
offer_end	The date and time the offer ends.	time
offer_start	The date and time the offer begins.	time
op_id	The ID number of the opportunity the quote is associated with.	integer
prod2q	The products associated with the quote.	pair
schedule_id	The ID number of the price schedule associated with the quote.	integer
sent	The date and time the quote was sent.	time
sent_to	The email address the quote was sent to.	string
status	The current status of the quote.	integer
template_id	The ID number of the template used in the quote.	integer
total	The total value of the quote.	integer
total_curr_id	The currency ID associated with the total value of the quote.	integer
total_rate_id	The exchange rate ID associated with the total value of the quote.	integer
updated_by	The ID number of the staff member who last updated the quote.	integer

Search API

The pairs described in the following table are available to use in search functions.

Table 25: Search Pairs

Name	Use	Type
search_args	The search argument.	pair

Table 25: Search Pairs (Continued)

Name	Use	Type
search_field	The search fields.	pair
name	The name of the run-time selectable filter being searched on.	string
compare_value	The value of the field being searched on.	string
view_id	The code number of the view being used to display results.	integer

SLA instance API

The pairs described in the following table are available to use in the SLA instance functions.

Table 26: SLA Instance Pairs

Name	Use	Type
activedate	The activation date of the SLA instance.	time
owner_id	The ID number of the contact or organization the SLA instance.	integer
owner_tbl	The type of owner. • 2—Contact • 3—Organization	integer
sla_id	The ID number of the SLA that the SLA instance is associated with.	integer

Task instance API

The pairs described in the following table are available to use in task instance functions.

Table 27: Task Instance Pairs

Name	Use	Type
assgn_acct_id	The ID number of the staff member assigned to the task instance.	integer
c_id	The ID number of the contact associated with the task instance.	integer
completed	The date and time the task instance was completed.	time
created_by	The ID number of the staff member the task instance was created by.	integer
custom_field	The custom fields associated with the task instance.	string
due_date	The date and time the task instance is due.	time
name	The name of the task instance.	string
notes	The notes associated with the task instance.	string
op_id	The opportunity the task instance is associated with.	integer
org_id	The organization the task instance is associated with.	integer
planned_completion	The planned completion date and time for the task instance.	time
prev_assgn_acct_id	The staff member who was previously assigned to the task instance.	integer
source	The creation source of the task instance. • 10—XML API • 19—Sales Console • 26—Marketing Campaign	integer

Table 27: Task Instance Pairs (Continued)

Name	Use	Type
task_id	The ID number of the task associated with the task instance.	integer
updated_by	The ID number of the last staff member to update the task instance.	integer

Appendix B
Source Codes

This appendix describes the source codes that can be used when creating contacts, incidents, opportunities, and organizations.

Contact API source codes

The following source codes are available when using the contact_create function.

Table 28: Contact API Source Codes

Code	Source
1	Support Console
2	Ask a Question Page
3	Email
4	Answer feedback
5	Site feedback
7	Rule
10	Public API
11	Live
12	Answer Console
13	Postal mail
14	Fax
15	Telephone
16	CSR (Other)
17	Marketing outbound email response
18	Marketing contact upload

Table 28: Contact API Source Codes

Code	Source
19	Sales Console
20	Marketing Contact Console
21	Internet (RightNow Sales)
22	Direct Mail (RightNow Sales)
23	Sales Representative (RightNow Sales)
24	Service (RightNow Sales)
25	Customer referral (RightNow Sales)
26	Marketing Campaign Console

Incident API source codes

The following source codes are available when using the incident_create function.

Table 29: Incident API Source Codes

Code	Source
1	Support Console
2	Personal Assistance (Ask a Question Page)
3	Email
4	Answer Feedback
5	Site Feedback
10	Public API
11	Live
13	Postal Mail
14	Fax
15	Phone

Table 29: Incident API Source Codes (Continued)

Code	Source
17	Marketing mailbox

Opportunity API source codes

The following source codes are available when using the sa_opp_create function.

Table 30: Opportunity API Source Codes

Code	Source
10	Public API
19	Sales Console
26	Marketing Campaign Console

Organization API source codes

The following source codes are available when using the org_create function.

Table 31: Organization API Source Codes

Code	Source
1	Support Console
2	Ask a Question Page
3	Email
4	Answer feedback
5	Site feedback
7	Rule
10	Public API
11	Live
12	Answer Console

Table 31: Organization API Source Codes (Continued)

Code	Source
13	Postal mail
14	Fax
15	Telephone
16	CSR (Other)
17	Marketing outbound email response
18	Marketing contact upload
19	Sales Console
20	Marketing Contact Console
21	Internet (RightNow Sales)
22	Direct Mail (RightNow Sales)
23	Sales Representative (RightNow Sales)
24	Service (RightNow Sales)
25	Customer referral (RightNow Sales)
26	Marketing Campaign Console

Appendix C
Logical Bit Flags

This appendix describes the logical bit flags that can be used when creating, updating, deleting, and retrieving answers, contacts, incidents, opportunities, and organizations.

You can use more than one logical bit flag for a function by adding the hex values of the flags together. For example, when using the incident_update function you can pass logical bit flags to execute an external event (CALL_EXTERNAL_EVENT—0x00002) and a send a staff notification (NOTIFY_STAFF—0x00008) by passing the hex value 0x0000A.

Note When using multiple logical bit flags, you must be doing your addition in hexidecimal.

Answer API

The following logical bit flags are available when performing answer functions.

Table 32: Answer Flags

Name	Hex Value	Use
CALL_EXTERNAL_ EVENT	0x00002	Specify whether an external event should be called if applicable. If this is not set, no external event will be executed.
NO_WORKFLOW	0x00800	Specify whether to apply business rules. If this is not set, rules will be triggered.

Contact API

The following logical bit flags are available when performing contact functions.

Table 33: Contact Flags

Name	Hex Value	Use
CALL_EXTERNAL_ EVENT	0x00002	Specify whether an external event should be called if applicable. If this is not set, no external event will be executed.
GET_THREADS	0x00200	Specify whether notes threads should be included in the return value when using the XML function contact_get. This flag only applies to contacts that are associated with a Sales state (sa_state=1.)
NO_WORKFLOW	0x00800	Specify whether to apply business rules. If this is not set, rules will be triggered.

Incident API

The following logical bit flags are available when performing incident functions.

Table 34: Incident Flags

Name	Hex Value	Use
CALL_EXTERNAL_ EVENT	0x00002	Specify whether an external event should be called if applicable. If this is not set, no external event will be executed.
GET_SEC_ CONTACTS	0x01000	Specify whether sec_contact IDs should be included in the return value when using the XML function incident_get.
GET_THREADS	0x00200	Specify whether incident threads should be included in the return value when using the XML function incident_get.

Table 34: Incident Flags (Continued)

Name	Hex Value	Use
MSG_RECEIPT	0x00010	Specify whether to send a receipt message to the address indicated in the incident contact information field upon incident creation. Used in incident_create.
NO_WORKFLOW	0x00800	Specify whether to apply business rules. If this is not set, rules will be triggered.
NOTIFY_STAFF	0x00008	Specify whether to send a notification email message to the staff member assigned to a new incident.

Opportunity API

The following flags are available when performing opportunity functions.

Table 35: Opportunity Flags

Name	Hex Value	Use
CALL_EXTERNAL_EVENT	0x00002	Specify whether an external event should be called if applicable. If this is not set, no external event will be executed.
GET_OPP2CONTACTS	0x01000	Specify whether to retrieve opportunity to contact mapping when using the XML function sa_opp_get.
GET_THREADS	0x00200	Specify whether opportunity threads should be included in the return value when using the XML function sa_opp_get.
NO_WORKFLOW	0x00800	Specify whether to apply business rules. If this is not set, rules will be triggered.

Organization API

The following logical bit flags are available when performing organization functions.

Table 36: Organization Flags

Name	Hex Value	Use
CALL_EXTERNAL_ EVENT	0x00002	Specify whether an external event should be called if applicable. If this is not set, no external event will be executed.
GET_ORG_ADDR	0x00400	Specify whether organization addresses should be included in the return value when using the XML function org_get.
GET_THREADS	0x00200	Specify whether notes threads should be included in the return value when using the XML function org_get. This flag only applies to organizations that are associated with a Sales state (sa_state=1.)
NO_WORKFLOW	0x00800	Specify whether to apply business rules. If this is not set, rules will be triggered.

Appendix D
Database Schema Tables

This appendix describes the schema tables of the database underlying RightNow CRM. The following list of tables are in alphabetical order and include the associated columns; the datatype including varchar2 (character), date, number, or clob (character byte objects); and the null value which indicates if the field is required.

To view and print the table structures for the *entire* RightNow CRM schema, open a command prompt and execute the following command from your utilities directory:

```
dbaudit -S <table_name> <interface_name>
```

To view and print the table structure for a *specific* table in the RightNow CRM schema, open a command prompt and execute the following command from your utilities directory:

```
dbaudit -s <table_name> <interface_name>
```

Schema tables

The following tables include attribute data for each RightNow CRM schema table.

Table 37: ac_alerts

Column Name	Datatype	Null
a_id	NUMBER	No
created	DATE	Yes
e_id	NUMBER	Yes
hit_count	NUMBER	Yes
hit_rule	NUMBER	Yes
last_fired	DATE	Yes
name	VARCHAR2	Yes
sch_id	NUMBER	Yes

Table 38: ac_cntr_tabs

Column Name	Datatype	Null
ac_id	NUMBER	Yes
ct_id	NUMBER	No
label	VARCHAR2	Yes

Table 39: ac_color_schemes

Column Name	Datatype	Null
color_trans	NUMBER	Yes
colors	VARCHAR2	Yes
cs_id	NUMBER	No
folder_id	NUMBER	Yes
internal	NUMBER	Yes
name	VARCHAR2	Yes

Table 40: ac_excepts

Column Name	Datatype	Null
attr	NUMBER	Yes
col_refs	VARCHAR2	Yes
data_type	NUMBER	No
e_id	NUMBER	No
except_opts	NUMBER	Yes

Table 40: ac_excepts (Continued)

Column Name	Datatype	Null
expr1	VARCHAR2	Yes
expr2	VARCHAR2	Yes
fi_id	NUMBER	Yes
graph_datamark	NUMBER	Yes
graph_line_color	VARCHAR2	Yes
hcol_rf	NUMBER	Yes
n_id	NUMBER	Yes
name	VARCHAR2	Yes
oper	NUMBER	Yes
tab_cell_color	VARCHAR2	Yes
tab_datamark	NUMBER	Yes
tab_font	VARCHAR2	Yes

Table 41: ac_fld_info

Column Name	Datatype	Null
align	NUMBER	No
col_rf	NUMBER	No
col_type	NUMBER	No
comp_opts	NUMBER	Yes
data_type	NUMBER	No
excl_intv	VARCHAR2	Yes
fi_id	NUMBER	No
format	VARCHAR2	Yes

Table 41: ac_fld_info (Continued)

Column Name	Datatype	Null
format_dec	NUMBER	Yes
format_opts	NUMBER	No
format_style	NUMBER	Yes
format_type	NUMBER	Yes
heading	VARCHAR2	Yes
max_len	NUMBER	Yes
n_id	NUMBER	No
opts	NUMBER	Yes
outp_def	VARCHAR2	Yes
seq	NUMBER	Yes
trend_frcst_units	NUMBER	Yes
trend_opts	NUMBER	Yes
width	VARCHAR2	Yes
wt_col_rf	NUMBER	Yes

Table 42: ac_graph_src

Column Name	Datatype	Null
fi_id	NUMBER	No
g_id	NUMBER	No
gsrc_id	NUMBER	No
stype	NUMBER	No

Table 43: ac_graph_styles

Column Name	Datatype	Null
bar_grad	NUMBER	Yes
bar_grad_clr	VARCHAR2	Yes
bar_grad_type	NUMBER	Yes
bar_opts	NUMBER	Yes
bar_outline_clr	VARCHAR2	Yes
bar_outline_type	NUMBER	Yes
bar_spacing	NUMBER	Yes
bar_width	NUMBER	Yes
bubble_size	NUMBER	Yes
bubble_type	NUMBER	Yes
clabel_bkclr	VARCHAR2	Yes
clabel_border_clr	VARCHAR2	Yes
clabel_border_type	NUMBER	Yes
clabel_font	VARCHAR2	Yes
clabel_shadow_clr	VARCHAR2	Yes
clabel_shadow_type	NUMBER	Yes
color_back_clr	VARCHAR2	Yes
color_back_trans	NUMBER	Yes
color_bkgd_clr	VARCHAR2	Yes
color_bkgd_grad_clr	VARCHAR2	Yes
color_bkgd_grad_type	NUMBER	Yes
color_bttm_clr	VARCHAR2	Yes

Table 43: ac_graph_styles (Continued)

Column Name	Datatype	Null
color_bttm_trans	NUMBER	Yes
color_side_clr	VARCHAR2	Yes
color_side_trans	NUMBER	Yes
color_theme	NUMBER	Yes
color_trans	NUMBER	Yes
colors	VARCHAR2	Yes
dlabel_bclr	VARCHAR2	Yes
dlabel_bkclr	VARCHAR2	Yes
dlabel_display	NUMBER	Yes
dlabel_font	VARCHAR2	Yes
dlabel_pos	NUMBER	Yes
elabel_bkclr	VARCHAR2	Yes
elabel_border_clr	VARCHAR2	Yes
elabel_border_type	NUMBER	Yes
elabel_font	VARCHAR2	Yes
elabel_pos	NUMBER	Yes
exc_box_bkclr	VARCHAR2	Yes
exc_box_border_clr	VARCHAR2	Yes
exc_box_border_type	NUMBER	Yes
exc_box_font	VARCHAR2	Yes
exc_box_pos	NUMBER	Yes
exc_box_shadow_clr	VARCHAR2	Yes
exc_box_shadow_type	NUMBER	Yes

Table 43: ac_graph_styles (Continued)

Column Name	Datatype	Null
exc_line_color_theme	NUMBER	Yes
exc_line_colors	VARCHAR2	Yes
exc_line_width	NUMBER	Yes
exc_note_bkclr	VARCHAR2	Yes
exc_note_border_clr	VARCHAR2	Yes
exc_note_border_show	NUMBER	Yes
exc_note_font	VARCHAR2	Yes
folder_id	NUMBER	Yes
gauge_color	VARCHAR2	Yes
gauge_color_tbl	VARCHAR2	Yes
gauge_font	VARCHAR2	Yes
gauge_opts	NUMBER	Yes
gauge_scale_clr	VARCHAR2	Yes
grid_cleft_clr	VARCHAR2	Yes
grid_cmajor_clr	VARCHAR2	Yes
grid_cminor_clr	VARCHAR2	Yes
grid_cright_clr	VARCHAR2	Yes
grid_czero_clr	VARCHAR2	Yes
grid_opts	NUMBER	Yes
grid_vbottom_clr	VARCHAR2	Yes
grid_vmajor_clr	VARCHAR2	Yes
grid_vminor_clr	VARCHAR2	Yes
grid_vtop_clr	VARCHAR2	Yes

Table 43: ac_graph_styles (Continued)

Column Name	Datatype	Null
grid_vzero_clr	VARCHAR2	Yes
gs_id	NUMBER	No
internal	NUMBER	Yes
label_opts	NUMBER	Yes
legend_bkclr	VARCHAR2	Yes
legend_border_clr	VARCHAR2	Yes
legend_border_type	NUMBER	Yes
legend_cols	NUMBER	Yes
legend_font	VARCHAR2	Yes
legend_layout	NUMBER	Yes
legend_pos	NUMBER	Yes
legend_shadow_clr	VARCHAR2	Yes
legend_shadow_type	NUMBER	Yes
legend_show	NUMBER	Yes
line_sym_type	NUMBER	Yes
line_width	NUMBER	Yes
name	VARCHAR2	Yes
pie_show_gaps	NUMBER	Yes
radar_trnsp	NUMBER	Yes
scale_bar_font	VARCHAR2	Yes
scale_line_font	VARCHAR2	Yes
scale_line_max	NUMBER	Yes
scale_line_min	NUMBER	Yes

Table 43: ac_graph_styles (Continued)

Column Name	Datatype	Null
scale_line_pctover	NUMBER	Yes
scale_major_tics	NUMBER	Yes
scale_max	NUMBER	Yes
scale_min	NUMBER	Yes
scale_minor_tics	NUMBER	Yes
scale_opts	NUMBER	Yes
scale_pctover	NUMBER	Yes
scale_rot_bar_lbls	NUMBER	Yes
scale_rot_line_lbls	NUMBER	Yes
threed_bdepth	NUMBER	Yes
threed_xoff	NUMBER	Yes
threed_yoff	NUMBER	Yes
title_bkclr	VARCHAR2	Yes
title_border_clr	VARCHAR2	Yes
title_border_type	NUMBER	Yes
title_font	VARCHAR2	Yes
title_pos	NUMBER	Yes
title_shadow_clr	VARCHAR2	Yes
title_shadow_type	NUMBER	Yes
title_show	NUMBER	Yes
vlabel_bkclr	VARCHAR2	Yes
vlabel_border_clr	VARCHAR2	Yes
vlabel_border_type	NUMBER	Yes

Table 43: ac_graph_styles (Continued)

Column Name	Datatype	Null
vlabel_font	VARCHAR2	Yes
vlabel_rotation	NUMBER	Yes
vlabel_shadow_clr	VARCHAR2	Yes
vlabel_shadow_type	NUMBER	Yes
xy_fill_area	NUMBER	Yes

Table 44: ac_graphs

Column Name	Datatype	Null
clabel	VARCHAR2	Yes
cseq	NUMBER	No
cspan	NUMBER	No
format	NUMBER	No
g_id	NUMBER	No
gs_id	NUMBER	No
gsub_type	NUMBER	No
gtype	NUMBER	No
height	NUMBER	No
n_id	NUMBER	No
rseq	NUMBER	No
rspan	NUMBER	No
sync_scales	NUMBER	Yes
title	VARCHAR2	Yes
vlabel1	VARCHAR2	Yes

Table 44: ac_graphs (Continued)

Column Name	Datatype	Null
vlabel2	VARCHAR2	Yes
width	NUMBER	No

Table 45: ac_nodes

Column Name	Datatype	Null
ac_id	NUMBER	No
delivery_style	NUMBER	No
format_opts	NUMBER	No
n_id	NUMBER	No
node_id	NUMBER	No
output_def	VARCHAR2	Yes
output_def_style	NUMBER	No
output_opts	NUMBER	No
row_limit	NUMBER	Yes
sel_crit_style	NUMBER	No
style_id	NUMBER	No
sub_title	VARCHAR2	Yes
title	VARCHAR2	Yes

Table 46: ac_sch_filters

Column Name	Datatype	Null
ac_id	NUMBER	Yes

Table 46: ac_sch_filters (Continued)

Column Name	Datatype	Null
col_rf	NUMBER	Yes
sch_id	NUMBER	Yes
sf_id	NUMBER	No
value	VARCHAR2	Yes

Table 47: ac_sch_sort

Column Name	Datatype	Null
as_id	NUMBER	No
disp_col	NUMBER	Yes
sch_id	NUMBER	Yes
seq	NUMBER	Yes
value	VARCHAR2	Yes

Table 48: ac_schedules

Column Name	Datatype	Null
ac_id	NUMBER	No
body	VARCHAR2	Yes
created	DATE	Yes
format	NUMBER	No
last_mod	DATE	Yes
last_run	DATE	Yes
name	VARCHAR2	Yes

Table 48: ac_schedules (Continued)

Column Name	Datatype	Null
next_run	DATE	Yes
sch_hours	NUMBER	No
sch_id	NUMBER	No
sch_mdays	NUMBER	No
sch_months	NUMBER	No
sch_opts	NUMBER	Yes
sch_wdays	NUMBER	No
subject	VARCHAR2	Yes

Table 49: ac_scripts

Column Name	Datatype	Null
exit_code	VARCHAR2	Yes
header_code	VARCHAR2	Yes
init_code	VARCHAR2	Yes
n_id	NUMBER	No
process_code	VARCHAR2	Yes
scr_id	NUMBER	No

Table 50: ac_style_attrs

Column Name	Datatype	Null
attr_type	NUMBER	Yes
border	VARCHAR2	Yes

Table 50: ac_style_attrs (Continued)

Column Name	Datatype	Null
cell_padding	VARCHAR2	Yes
cell_spacing	NUMBER	Yes
color	VARCHAR2	Yes
css	VARCHAR2	Yes
font	VARCHAR2	Yes
s_id	NUMBER	Yes
sa_id	NUMBER	No
seq	NUMBER	Yes

Table 51: ac_styles

Column Name	Datatype	Null
folder_id	NUMBER	Yes
gs_id	NUMBER	Yes
internal	NUMBER	Yes
name	VARCHAR2	Yes
s_id	NUMBER	No

Table 52: account_speed_dial

Column Name	Datatype	Null
acct_id	NUMBER	Yes
name	VARCHAR2	Yes
phone	VARCHAR2	Yes

Table 53: accounts

Column Name	Datatype	Null
acct_id	NUMBER	No
acd_group	VARCHAR2	Yes
acd_passwd	VARCHAR2	Yes
alt_first_name	VARCHAR	Yes
alt_last_name	VARCHAR	Yes
attr	NUMBER	Yes
country_id	NUMBER	Yes
dca_enabled	NUMBER	Yes
def_currency	NUMBER	No
display_name	VARCHAR2	Yes
eas_id	NUMBER	Yes
email_address	VARCHAR2	Yes
email_notif	NUMBER	Yes
first_name	VARCHAR2	Yes
group_id	NUMBER	Yes
invalid_logins	NUMBER	Yes
last_name	VARCHAR2	Yes
last_sync	DATE	Yes
login	VARCHAR2	Yes
lvl10_id	NUMBER	Yes
lvl11_id	NUMBER	Yes
lvl12_id	NUMBER	Yes

Table 53: accounts (Continued)

Column Name	Datatype	Null
lvl1_id	NUMBER	Yes
lvl2_id	NUMBER	Yes
lvl3_id	NUMBER	Yes
lvl4_id	NUMBER	Yes
lvl5_id	NUMBER	Yes
lvl6_id	NUMBER	Yes
lvl7_id	NUMBER	Yes
lvl8_id	NUMBER	Yes
lvl9_id	NUMBER	Yes
notif_pending	NUMBER	No
password	VARCHAR2	Yes
password_exp	DATE	Yes
password_history	VARCHAR	Yes
phone	VARCHAR2	Yes
profile_id	NUMBER	Yes
sa_def_cmode	NUMBER	No
seq	NUMBER	Yes
sessionid	VARCHAR2	Yes
sid_esync	VARCHAR	Yes
sid_isync	NUMBER	Yes
sid_wap	NUMBER	Yes
signature	VARCHAR2	Yes
softphone	NUMBER	Yes

Table 53: accounts (Continued)

Column Name	Datatype	Null
start_console	NUMBER	Yes
sync_sessid	VARCHAR2	Yes
sync_state	NUMBER	Yes
terr_id	NUMBER	Yes
timezone	NUMBER	Yes

Table 54: agent_acd_modes

Column Name	Datatype	Null
aam_id	NUMBER	No
cti_mode	NUMBER	Yes
notes	VARCHAR2	Yes
rnt_mode	NUMBER	Yes
seq	NUMBER	Yes
switch_type	NUMBER	Yes

Table 55: analytics_core

Column Name	Datatype	Null
ac_id	NUMBER	No
ac_public	NUMBER	Yes
ac_type	NUMBER	No
act_ac_id	NUMBER	Yes
cdate_offset	VARCHAR2	Yes

Table 55: analytics_core (Continued)

Column Name	Datatype	Null
comp_exist	NUMBER	Yes
created	DATE	Yes
created_by	NUMBER	Yes
cseq	NUMBER	Yes
cspan	NUMBER	Yes
ct_id	NUMBER	Yes
delivery_style	NUMBER	Yes
format_opts	NUMBER	Yes
interface_id	NUMBER	No
internal	NUMBER	Yes
last_gen	DATE	Yes
last_gen_by	NUMBER	Yes
last_mod	DATE	Yes
last_mod_by	NUMBER	Yes
module	NUMBER	Yes
name	VARCHAR2	Yes
notes	VARCHAR2	Yes
output_def	VARCHAR2	Yes
output_def_style	NUMBER	Yes
owner	NUMBER	Yes
parent_id	NUMBER	Yes
rseq	NUMBER	Yes
rspan	NUMBER	Yes

Table 55: analytics_core (Continued)

Column Name	Datatype	Null
sel_crit_style	NUMBER	Yes
seq	NUMBER	Yes
style_id	NUMBER	Yes
sub_title	VARCHAR2	Yes
sys_exe_path	VARCHAR2	Yes
sys_fltr_path	VARCHAR2	Yes
title	VARCHAR2	Yes
view_id	NUMBER	Yes

Table 56: ans_access

Column Name	Datatype	Null
access_id	NUMBER	No
rank	NUMBER	No

Table 57: ans_notif

Column Name	Datatype	Null
a_id	NUMBER	No
c_id	NUMBER	No
interface_id	NUMBER	Yes
start_time	DATE	Yes

Table 58: ans_phrases

Column Name	Datatype	Null
a_id	NUMBER	No
access_mask	VARCHAR	Yes
lang_id	NUMBER	No
natt	NUMBER	No
ncat	NUMBER	No
ndesc	NUMBER	No
nflds	NUMBER	No
nkeyw	NUMBER	No
nprod	NUMBER	No
nsol	NUMBER	No
nsum	NUMBER	No
status_type	NUMBER	No
top_lvl	NUMBER	Yes
word	VARCHAR2	No
word_count	NUMBER	No

Table 59: ans_stats

Column Name	Datatype	Null
a_id	NUMBER	No
hits	NUMBER	Yes
interface_id	NUMBER	Yes

Table 59: ans_stats (Continued)

Column Name	Datatype	Null
solved_1	NUMBER	Yes
solved_2	NUMBER	Yes
solved_3	NUMBER	Yes
solved_4	NUMBER	Yes
solved_5	NUMBER	Yes
stat_date	DATE	Yes

Table 60: answers

Column Name	Datatype	Null
a_id	NUMBER	No
access_mask	VARCHAR2	Yes
assgn_acct_id	NUMBER	Yes
assgn_group_id	NUMBER	Yes
created	DATE	Yes
description	CLOB	Yes
doc_size	NUMBER	Yes
doc_type	VARCHAR2	Yes
doc_url	VARCHAR2	Yes
escldate	DATE	Yes
escllevel	NUMBER	Yes
expires	DATE	Yes
keywords	VARCHAR2	Yes
lang_id	NUMBER	No

Table 60: answers (Continued)

Column Name	Datatype	Null
last_access	DATE	Yes
last_edited_by	NUMBER	Yes
last_notify	DATE	Yes
m_id	NUMBER	Yes
next_notify	DATE	Yes
notes	VARCHAR2	Yes
publish_on	DATE	Yes
rule_state	NUMBER	Yes
solution	CLOB	Yes
solved_count	NUMBER	No
static_solved	NUMBER	Yes
status_id	NUMBER	No
status_type	NUMBER	No
summary	VARCHAR2	No
updated	DATE	Yes

Table 61: archived_incidents

Column Name	Datatype	Null
c_id	NUMBER	Yes
closed	DATE	Yes
created	DATE	Yes
file_dir	VARCHAR2	Yes
i_id	NUMBER	No

Table 61: archived_incidents (Continued)

Column Name	Datatype	Null
interface_id	NUMBER	Yes
last_resp	DATE	Yes
ref_no	VARCHAR2	No
title	VARCHAR2	Yes

Table 62: billable_tasks

Column Name	Datatype	Null
seq	NUMBER	Yes
task_id	NUMBER	No

Table 63: call_activity

Column Name	Datatype	Null
acct_id	NUMBER	Yes
activity	NUMBER	No
cti_call_id	NUMBER	No
dial_num	VARCHAR2	Yes
end_time	DATE	Yes
start_time	DATE	No

Table 64: clicktrack

Column Name	Datatype	Null
c_id	NUMBER	Yes

Table 64: clicktrack (Continued)

Column Name	Datatype	Null
interface_id	NUMBER	No
parm	VARCHAR2	Yes
sessionid	VARCHAR2	No
timestamp	DATE	Yes
val	VARCHAR2	Yes

Table 65: cluster_class

Column Name	Datatype	Null
frequency	NUMBER	No
interface_id	NUMBER	No
node_id	NUMBER	No
weight	NUMBER	No
word	VARCHAR2	No

Table 66: cluster_info

Column Name	Datatype	Null
id	NUMBER	No
interface_id	NUMBER	No
solved_count	NUMBER	Yes
summary	VARCHAR2	Yes

Table 67: cluster_tree

Column Name	Datatype	Null
info_ptr	NUMBER	Yes
interface_id	NUMBER	No
leaf_cnt	NUMBER	Yes
node_id	NUMBER	No
parent	NUMBER	Yes
ptr_type	NUMBER	Yes
strength	NUMBER	Yes
tot_leaf_cnt	NUMBER	Yes

Table 68: configuration

Column Name	Datatype	Null
incident_date	DATE	No
last_ans_id	NUMBER	Yes
last_contact_id	NUMBER	No
last_incident_id	NUMBER	No
last_meta_ans_id	NUMBER	Yes
last_opp_id	NUMBER	Yes
last_org_id	NUMBER	Yes
last_proof_id	NUMBER	Yes
last_quote_id	NUMBER	Yes
last_ref_no_id	VARCHAR2	No

Table 68: configuration (Continued)

Column Name	Datatype	Null
last_rnl_cache	DATE	Yes
last_rule_id	NUMBER	No
last_sp_cache	DATE	Yes
last_ss_cache	DATE	Yes
last_state_id	NUMBER	No
last_ti_id	NUMBER	Yes
options	NUMBER	No
rnw_db_upgrade_level	NUMBER	Yes
rnw_db_version	NUMBER	Yes

Table 69: contact_types

Column Name	Datatype	Null
ctype_id	NUMBER	No
seq	NUMBER	Yes

Table 70: contact_sessions

Column Name	Datatype	Null
c_id	NUMBER	No
has_cookie	NUMBER	Yes
interface_id	NUMBER	No
login_time	NUMBER	Yes
logout_time	NUMBER	Yes

Table 70: contact_sessions (Continued)

Column Name	Datatype	Null
session_start_time	NUMBER	Yes
sessionid	VARCHAR	No

Table 71: contacts

Column Name	Datatype	Null
alt_first_name	VARCHAR	Yes
alt_last_name	VARCHAR	Yes
c_id	NUMBER	No
cat_lvl1	NUMBER	Yes
cat_lvl2	NUMBER	Yes
cat_lvl3	NUMBER	Yes
cat_lvl4	NUMBER	Yes
cat_lvl5	NUMBER	Yes
cat_lvl6	NUMBER	Yes
cert	VARCHAR2	Yes
cert_alt1	VARCHAR2	Yes
cert_alt2	VARCHAR2	Yes
city	VARCHAR2	Yes
country_id	NUMBER	Yes
created	DATE	Yes
css_state	NUMBER	No
ctype_id	NUMBER	Yes
disabled	NUMBER	No

Table 71: contacts (Continued)

Column Name	Datatype	Null
email	VARCHAR2	Yes
email_alt1	VARCHAR2	Yes
email_alt2	VARCHAR2	Yes
email_invalid	NUMBER	Yes
first_name	VARCHAR2	Yes
last_name	VARCHAR2	Yes
lines_per_page	NUMBER	Yes
login	VARCHAR2	Yes
lvl10_id	NUMBER	Yes
lvl11_id	NUMBER	Yes
lvl12_id	NUMBER	Yes
lvl1_id	NUMBER	Yes
lvl2_id	NUMBER	Yes
lvl3_id	NUMBER	Yes
lvl4_id	NUMBER	Yes
lvl5_id	NUMBER	Yes
lvl6_id	NUMBER	Yes
lvl7_id	NUMBER	Yes
lvl8_id	NUMBER	Yes
lvl9_id	NUMBER	Yes
ma_alt_org_name	VARCHAR2	Yes
ma_mail_type	NUMBER	Yes
ma_opt_in	NUMBER	Yes

Table 71: contacts (Continued)

Column Name	Datatype	Null
ma_org_name	VARCHAR2	Yes
ma_state	NUMBER	No
org_id	NUMBER	Yes
password	VARCHAR2	Yes
ph_asst	VARCHAR2	Yes
ph_fax	VARCHAR2	Yes
ph_home	VARCHAR2	Yes
ph_mobile	VARCHAR2	Yes
ph_office	VARCHAR2	Yes
postal_code	VARCHAR2	Yes
prod_lvl1	NUMBER	Yes
prod_lvl2	NUMBER	Yes
prod_lvl3	NUMBER	Yes
prod_lvl4	NUMBER	Yes
prod_lvl5	NUMBER	Yes
prod_lvl6	NUMBER	Yes
prov_id	NUMBER	Yes
rule_state	NUMBER	Yes
sa_state	NUMBER	No
search_text	VARCHAR2	Yes
search_type	NUMBER	Yes
sessionid	VARCHAR2	Yes
source	NUMBER	Yes

Table 71: contacts (Continued)

Column Name	Datatype	Null
street	VARCHAR2	Yes
title	VARCHAR2	Yes
updated	DATE	Yes

Table 72: countries

Column Name	Datatype	Null
abrev	VARCHAR2	Yes
country_id	NUMBER	No
phone_code	NUMBER	Yes
phone_mask	VARCHAR2	Yes
postal_mask	VARCHAR2	Yes
seq	NUMBER	Yes

Table 73: cti_calls

Column Name	Datatype	Null
acct_id	NUMBER	Yes
acd_flag	NUMBER	Yes
ani	VARCHAR2	Yes
app_data	VARCHAR2	Yes
cti_call_id	NUMBER	No
end_time	DATE	Yes
start_time	DATE	Yes

Table 74: cti_current_call

Column Name	Datatype	Null
calldata	VARCHAR2	Yes
cti_call_id	NUMBER	No

Table 75: cti_login

Column Name	Datatype	Null
acct_id	NUMBER	Yes
acd_group	NUMBER	Yes
agent_extension	NUMBER	Yes
cti_login_id	NUMBER	No
end_time	DATE	Yes
start_time	DATE	Yes

Table 76: cti_mode_changes

Column Name	Datatype	Null
acd_mode	NUMBER	No
cti_call_id	NUMBER	Yes
cti_login_id	NUMBER	No
end_time	DATE	Yes
start_time	DATE	Yes

Table 77: cua_contacts

Column Name	Datatype	Null
c_id	NUMBER	Yes
city	VARCHAR2	Yes
country	VARCHAR2	Yes
cua_c_id	NUMBER	No
email	VARCHAR2	Yes
first_name	VARCHAR2	Yes
last_name	VARCHAR2	Yes
login	VARCHAR2	Yes
org_id	NUMBER	Yes
postal_code	VARCHAR2	Yes
province	VARCHAR2	Yes
street	VARCHAR2	Yes

Table 78: currencies

Column Name	Datatype	Null
abbreviation	VARCHAR2	No
currency_id	NUMBER	No
dec_char	VARCHAR2	No
decimal_precision	NUMBER	No
disp_opts	NUMBER	No
group_char	VARCHAR2	No

Table 78: currencies (Continued)

Column Name	Datatype	Null
notes	VARCHAR2	Yes
seq	NUMBER	No
symbol	VARCHAR2	No

Table 79: custom_fields

Column Name	Datatype	Null
cf_id	NUMBER	No
cfgroup_id	NUMBER	Yes
col_name	VARCHAR2	Yes
data_type	NUMBER	Yes
default_value	VARCHAR2	Yes
field_size	NUMBER	Yes
indexed	NUMBER	Yes
mask	VARCHAR2	Yes
notes	VARCHAR2	Yes
required	NUMBER	Yes
seq	NUMBER	Yes
tbl	NUMBER	No

Table 80: dates

Column Name	Datatype	Null
interval_start	NUMBER	No

Table 80: dates (Continued)

Column Name	Datatype	Null
start_date	DATE	PRIMARY_KEY

Table 81: db_maint_hist

Column Name	Datatype	Null
end_db_upgrade_level	NUMBER	Yes
end_db_version	NUMBER	Yes
end_result	NUMBER	No
end_time	DATE	Yes
software_version	VARCHAR2	No
start_db_upgrade_level	NUMBER	No
start_db_version	NUMBER	No
start_time	DATE	Yes

Table 82: deleted_recs

Column Name	Datatype	Null
deleted	DATE	No
deleted_by	NUMBER	Yes
id	NUMBER	No
lbl	VARCHAR	Yes
sourc	VARCHAR2	Yes
tbl	NUMBER	No

Table 83: dependencies

Column Name	Datatype	Null
dep_id	NUMBER	No
dep_tbl	NUMBER	No
ref_id	NUMBER	No
ref_name	VARCHAR2	Yes
ref_subid	NUMBER	Yes
ref_tbl	NUMBER	No

Table 84: dictionary

Column Name	Datatype	Null
lang_id	NUMBER	No
stemword	VARCHAR2	No
wcnt	NUMBER	Yes
word	VARCHAR2	No

Table 85: exchange_rates

Column Name	Datatype	Null
from_currency	NUMBER	No
rate	NUMBER	No
rate_end	DATE	Yes
rate_id	NUMBER	No

Table 85: exchange_rates (Continued)

Column Name	Datatype	Null
rate_start	DATE	Yes
to_currency	NUMBER	No

Table 86: fattach

Column Name	Datatype	Null
content_type	VARCHAR2	Yes
created	DATE	Yes
descr	VARCHAR2	Yes
disabled	NUMBER	No
file_id	NUMBER	No
folder_id	NUMBER	Yes
id	NUMBER	No
idx	NUMBER	Yes
interface_id	NUMBER	Yes
internal	NUMBER	No
localfname	VARCHAR2	No
name	VARCHAR2	Yes
sz	NUMBER	Yes
tbl	NUMBER	No
updated	DATE	Yes
userfname	VARCHAR2	Yes

Table 87: gap_info

Column Name	Datatype	Null
answer_id	NUMBER	Yes
answer_score	NUMBER	Yes
cohesion	NUMBER	Yes
gap_id	NUMBER	No
gap_score	NUMBER	Yes
interface_id	NUMBER	No
num_inc	NUMBER	Yes
summary	VARCHAR2	Yes
urg_lev	NUMBER	Yes

Table 88: gap_tree

Column Name	Datatype	Null
gap_id	NUMBER	No
i_id	NUMBER	No
interface_id	NUMBER	No
score	NUMBER	Yes

Table 89: hier_folders

Column Name	Datatype	Null
folder_id	NUMBER	No

Table 89: hier_folders (Continued)

Column Name	Datatype	Null
parent_id	NUMBER	Yes
seq	NUMBER	Yes
sub_tbl	NUMBER	Yes
tbl	NUMBER	No

Table 90: hier_menus

Column Name	Datatype	Null
id	NUMBER	No
lvl	NUMBER	No
parent_id	NUMBER	Yes
seq	NUMBER	No
tbl	NUMBER	No

Table 91: holidays

Column Name	Datatype	Null
code	NUMBER	No
day	NUMBER	Yes
month	NUMBER	Yes
name	VARCHAR2	No
seq	NUMBER	Yes
year	NUMBER	Yes

Table 92: inc_performance

Column Name	Datatype	Null
acct_id	NUMBER	Yes
action_cnt	NUMBER	Yes
group_id	NUMBER	Yes
i_id	NUMBER	Yes
interface_id	NUMBER	Yes
intv_type	NUMBER	Yes
queue_id	NUMBER	Yes
rel_time	NUMBER	Yes
solved	NUMBER	Yes
source	NUMBER	Yes
time_end	DATE	Yes
time_start	DATE	Yes

Table 93: incidents

Column Name	Datatype	Null
assgn_acct_id	NUMBER	Yes
assgn_group_id	NUMBER	Yes
c_id	NUMBER	Yes
cat_lvl1	NUMBER	Yes
cat_lvl2	NUMBER	Yes
cat_lvl3	NUMBER	Yes

Table 93: incidents (Continued)

Column Name	Datatype	Null
cat_lvl4	NUMBER	Yes
cat_lvl5	NUMBER	Yes
cat_lvl6	NUMBER	Yes
closed	DATE	Yes
created	DATE	Yes
created_by	NUMBER	Yes
disp_lvl1	NUMBER	Yes
disp_lvl2	NUMBER	Yes
disp_lvl3	NUMBER	Yes
disp_lvl4	NUMBER	Yes
disp_lvl5	NUMBER	Yes
disp_lvl6	NUMBER	Yes
dormant	NUMBER	No
ei_cust	NUMBER	Yes
ei_staff	NUMBER	Yes
escldate	DATE	Yes
escllevel	NUMBER	Yes
i_id	NUMBER	No
initial_soln	DATE	Yes
interface_id	NUMBER	Yes
lang_id	NUMBER	Yes
last_resp	DATE	Yes
ma_mailing_id	NUMBER	Yes

Table 93: incidents (Continued)

Column Name	Datatype	Null
mailbox_id	NUMBER	Yes
org_id	NUMBER	Yes
prod_lvl1	NUMBER	Yes
prod_lvl2	NUMBER	Yes
prod_lvl3	NUMBER	Yes
prod_lvl4	NUMBER	Yes
prod_lvl5	NUMBER	Yes
prod_lvl6	NUMBER	Yes
queue_id	NUMBER	Yes
ref_no	VARCHAR2	No
rel_due	DATE	Yes
resp_sav	CLOB	Yes
rnl_queue_id	NUMBER	Yes
rr_id	NUMBER	Yes
rule_state	NUMBER	Yes
sessionid	VARCHAR2	Yes
sla_resp_delta	NUMBER	Yes
sla_rsln_delta	NUMBER	Yes
slai_id	NUMBER	Yes
source	NUMBER	Yes
status_id	NUMBER	No
status_type	NUMBER	No
subject	VARCHAR2	Yes

Table 93: incidents (Continued)

Column Name	Datatype	Null
updated	DATE	Yes
use_smime	NUMBER	No

Table 94: integration_errors

Column Name	Datatype	Null
acct_id	NUMBER	Yes
closed	DATE	Yes
cnt	NUMBER	Yes
created	DATE	Yes
error_grp	VARCHAR2	Yes
error_id	NUMBER	No
external_id	VARCHAR2	Yes
id	NUMBER	Yes
integration_msg	CLOB	Yes
interface_id	NUMBER	No
notified	DATE	Yes
result	CLOB	Yes
status	NUMBER	Yes
tbl	NUMBER	No
type	NUMBER	Yes
updated	DATE	Yes

Table 95: interfaces

Column Name	Datatype	Null
ans_churn	NUMBER	No
ans_misclassify	NUMBER	No
display_name	VARCHAR2	Yes
interface_id	NUMBER	No
lang_id	NUMBER	No
last_account_notify	DATE	Yes
last_cua_model	DATE	Yes
last_datamine	DATE	Yes
last_gap	DATE	Yes
last_notify	DATE	Yes
name	VARCHAR2	No
pur_prod_churn	NUMBER	No

Table 96: isync_recs

Column Name	Datatype	Null
acct_id	NUMBER	No
id	NUMBER	Yes
tbl	NUMBER	Yes

Table 97: keyword_searches

Column Name	Datatype	Null
interface_id	NUMBER	No
result_list	VARCHAR2	Yes
results	NUMBER	No
search_count	NUMBER	No
search_date	DATE	Yes
source_page	NUMBER	No
srch	VARCHAR2	No
stem	VARCHAR2	No
word_cnt	NUMBER	No

Table 98: labels

Column Name	Datatype	Null
fld	NUMBER	No
label	VARCHAR2	Yes
label_id	NUMBER	No
lang_id	NUMBER	No
tbl	NUMBER	No

Table 99: languages

Column Name	Datatype	Null
lang_id	NUMBER	No
name	VARCHAR2	No

Table 100: links

Column Name	Datatype	Null
access_time	DATE	Yes
from_node	VARCHAR2	No
static_strength	NUMBER	Yes
strength	NUMBER	No
to_node	NUMBER	No

Table 101: locked_folders

Column Name	Datatype	Null
acct_id	NUMBER	Yes
folder_id	NUMBER	No
interface_id	NUMBER	Yes
internal	NUMBER	Yes
module	NUMBER	Yes
name	VARCHAR2	No
parent_id	NUMBER	Yes

Table 101: locked_folders (Continued)

Column Name	Datatype	Null
seq	NUMBER	Yes
sub_tbl	NUMBER	Yes
tbl	NUMBER	No

Table 102: locks

Column Name	Datatype	Null
acct_id	NUMBER	Yes
created	DATE	Yes
type	NUMBER	No

Table 103: ma_bounced_msgs

Column Name	Datatype	Null
c_id	NUMBER	No
created	DATE	Yes
format_id	NUMBER	No
from_addr	VARCHAR2	No
html	CLOB	Yes
mailing_id	NUMBER	No
reply_to_addr	VARCHAR2	Yes
retries	NUMBER	No
return_path_addr	VARCHAR2	No
subject	VARCHAR2	No

Table 103: ma_bounced_msgs (Continued)

Column Name	Datatype	Null
text	CLOB	Yes
to_addr	VARCHAR2	No
tracking_str	VARCHAR2	No

Table 104: ma_campaigns

Column Name	Datatype	Null
actual_cost	NUMBER	Yes
actual_cost_curr_id	NUMBER	Yes
actual_cost_rate_id	NUMBER	Yes
actual_leads	NUMBER	Yes
actual_opps	NUMBER	Yes
actual_sales	NUMBER	Yes
actual_sales_curr_id	NUMBER	Yes
actual_sales_rate_id	NUMBER	Yes
assgn_acct_id	NUMBER	Yes
budget	NUMBER	Yes
budget_curr_id	NUMBER	Yes
budget_rate_id	NUMBER	Yes
campaign_id	NUMBER	No
created	DATE	Yes
created_by	NUMBER	Yes
end_date	DATE	Yes
exp_cost	NUMBER	Yes

Table 104: ma_campaigns (Continued)

Column Name	Datatype	Null
exp_cost_curr_id	NUMBER	Yes
exp_cost_rate_id	NUMBER	Yes
exp_leads	NUMBER	Yes
exp_opps	NUMBER	Yes
exp_sales	NUMBER	Yes
exp_sales_curr_id	NUMBER	Yes
exp_sales_rate_id	NUMBER	Yes
folder_id	NUMBER	Yes
interface_id	NUMBER	No
name	VARCHAR2	Yes
notes	VARCHAR2	Yes
obj_summary	VARCHAR2	Yes
start_date	DATE	Yes
status_id	NUMBER	No
updated	DATE	Yes
updated_by	NUMBER	Yes

Table 105: ma_contact2list

Column Name	Datatype	Null
c_id	NUMBER	No
list_id	NUMBER	No

Table 106: ma_documents

Column Name	Datatype	Null
approved_by	NUMBER	Yes
created	DATE	Yes
created_by	NUMBER	Yes
disabled	NUMBER	No
doc_id	NUMBER	No
folder_id	NUMBER	Yes
html_xml	CLOB	Yes
interface_id	NUMBER	Yes
name	VARCHAR2	No
text_xml	CLOB	Yes
updated	DATE	Yes
updated_by	NUMBER	Yes
vis_email	NUMBER	Yes
vis_web	NUMBER	Yes

Table 107: ma_exclusions

Column Name	Datatype	Null
c_id	NUMBER	No
mailing_id	NUMBER	Yes
type	NUMBER	No

Table 108: ma_events

Column Name	Datatype	Null
c_id	NUMBER	Yes
campaign_id	NUMBER	Yes
created	DATE	Yes
filter_view_id	NUMBER	Yes
mailing_id	NUMBER	Yes
node_id	NUMBER	Yes
scheduled	DATE	Yes

Table 109: ma_formats

Column Name	Datatype	Null
created	DATE	Yes
created_by	NUMBER	Yes
doc_id	NUMBER	Yes
final_flag	NUMBER	Yes
format_id	NUMBER	No
from_addr_xml	VARCHAR2	Yes
from_name_xml	VARCHAR2	Yes
from_trans	NUMBER	Yes
last_launched	DATE	Yes
last_launched_by	NUMBER	Yes
last_proof_msg	VARCHAR	Yes

Table 109: ma_formats (Continued)

Column Name	Datatype	Null
list_id	NUMBER	Yes
mailbox_id	NUMBER	Yes
mailing_id	NUMBER	No
name	VARCHAR2	Yes
num_bounced	NUMBER	No
num_click_thru	NUMBER	No
num_excluded	NUMBER	Yes
num_replied	NUMBER	No
num_sent	NUMBER	No
num_unique_click_thru	NUMBER	No
num_unique_viewed	NUMBER	No
num_unsubs	NUMBER	No
num_viewed	NUMBER	No
percentage	NUMBER	Yes
proof_list_id	NUMBER	Yes
quantity	NUMBER	Yes
reply_addr_xml	VARCHAR2	Yes
reply_name_xml	VARCHAR2	Yes
reply_trans	NUMBER	Yes
scheduled	DATE	Yes
status_id	NUMBER	Yes
subject_trans	NUMBER	Yes
subject_xml	VARCHAR2	Yes

Table 109: ma_formats (Continued)

Column Name	Datatype	Null
updated	DATE	Yes
updated_by	NUMBER	Yes

Table 110: ma_import_templates

Column Name	Datatype	Null
field_map	VARCHAR2	Yes
template_id	NUMBER	No

Table 111: ma_imports

Column Name	Datatype	Null
completed	DATE	Yes
created_by	NUMBER	Yes
exist_mode	NUMBER	Yes
folder_id	NUMBER	Yes
force_import	NUMBER	Yes
import_id	NUMBER	No
imported_by	NUMBER	Yes
kimport_params	VARCHAR2	Yes
list_id	NUMBER	Yes
name	VARCHAR2	Yes
notes	VARCHAR2	Yes
num_error	NUMBER	Yes

Table 111: ma_imports (Continued)

Column Name	Datatype	Null
num_exist	NUMBER	Yes
num_insert	NUMBER	Yes
scheduled	DATE	Yes
seq	NUMBER	No
started	DATE	Yes
status_id	NUMBER	Yes
use_api	NUMBER	Yes

Table 112: ma_links

Column Name	Datatype	Null
cat_lvl1	NUMBER	Yes
cat_lvl2	NUMBER	Yes
cat_lvl3	NUMBER	Yes
cat_lvl4	NUMBER	Yes
cat_lvl5	NUMBER	Yes
cat_lvl6	NUMBER	Yes
disabled	NUMBER	No
folder_id	NUMBER	Yes
interface_id	NUMBER	Yes
link_id	NUMBER	No
name	VARCHAR2	No
notes	VARCHAR2	Yes
type	NUMBER	No

Table 112: ma_links (Continued)

Column Name	Datatype	Null
url	VARCHAR2	No

Table 113: ma_lists

Column Name	Datatype	Null
created_by	NUMBER	Yes
folder_id	NUMBER	Yes
interface_id	NUMBER	Yes
last_count	NUMBER	Yes
last_counted	DATE	Yes
list_id	NUMBER	No
name	VARCHAR2	No
notes	VARCHAR2	Yes
type	NUMBER	No
updated_by	NUMBER	Yes

Table 114: ma_mailing2cg

Column Name	Datatype	Null
exclude	NUMBER	No
list_id	NUMBER	Yes
mailing_id	NUMBER	No
seg_id	NUMBER	Yes

Table 115: ma_map2state

Column Name	Datatype	Null
campaign_id	NUMBER	Yes
mailing_id	NUMBER	Yes
node_id	NUMBER	Yes
state_id	NUMBER	Yes
wf_id	NUMBER	Yes

Table 116: ma_mailings

Column Name	Datatype	Null
campaign_id	NUMBER	Yes
cat_lvl1	NUMBER	Yes
cat_lvl2	NUMBER	Yes
cat_lvl3	NUMBER	Yes
cat_lvl4	NUMBER	Yes
cat_lvl5	NUMBER	Yes
cat_lvl6	NUMBER	Yes
created	DATE	Yes
ignore_suppression	NUMBER	Yes
interface_id	NUMBER	Yes
last_count	NUMBER	Yes
last_counted	DATE	Yes
mailing_id	NUMBER	No

Table 116: ma_mailings (Continued)

Column Name	Datatype	Null
name	VARCHAR2	No
notes	VARCHAR2	Yes
planned	DATE	Yes
query_string	VARCHAR2	Yes
scheduled	DATE	Yes
status_id	NUMBER	Yes
target_rate	NUMBER	Yes
type	NUMBER	No
updated	DATE	Yes

Table 117: ma_proof_content

Column Name	Datatype	Null
accept	NUMBER	Yes
c_id	NUMBER	Yes
comments	VARCHAR	Yes
proof_id	NUMBER	Yes

Table 118: ma_proof_trans

Column Name	Datatype	Null
c_id	NUMBER	Yes
campaign_id	NUMBER	Yes
created	TIME	Yes

Table 118: ma_proof_trans (Continued)

Column Name	Datatype	Null
doc_id	NUMBER	Yes
format_id	NUMBER	Yes
mailing_id	NUMBER	Yes
media	NUMBER	Yes
proof_id	NUMBER	Yes
proof_msg	VARCHAR	Yes
type	NUMBER	Yes

Table 119: ma_query_cache

Column Name	Datatype	Null
doc_id	NUMBER	Yes
format_id	NUMBER	Yes
query_cache	CLOB	No
view_id	NUMBER	Yes

Table 120: ma_segments

Column Name	Datatype	Null
created	DATE	Yes
created_by	NUMBER	Yes
folder_id	NUMBER	Yes
interface_id	NUMBER	Yes
last_count	NUMBER	Yes

Table 120: ma_segments (Continued)

Column Name	Datatype	Null
last_counted	DATE	Yes
name	VARCHAR2	No
notes	VARCHAR2	Yes
seg_id	NUMBER	No
updated	DATE	Yes
updated_by	NUMBER	Yes
view_id	NUMBER	No

Table 121: ma_start_nodes

Column Name	Datatype	Null
campaign_id	NUMBER	No
create_trans	NUMBER	No
name	VARCHAR	No
node_id	NUMBER	No
shortcut	VARCHAR	No

Table 122: ma_suppression

Column Name	Datatype	Null
email	VARCHAR	No
type	NUMBER	No

Table 123: ma_tags

Column Name	Datatype	Null
created	DATE	Yes
id	NUMBER	Yes
tag_id	NUMBER	Yes
tag_type	NUMBER	No
tbl	NUMBER	No

Table 124: ma_trans

Column Name	Datatype	Null
c_id	NUMBER	Yes
campaign_id	NUMBER	Yes
cf_id	NUMBER	Yes
created	DATE	Yes
doc_id	NUMBER	No
file_id	NUMBER	Yes
format_id	NUMBER	Yes
link_id	NUMBER	Yes
mailing_id	NUMBER	Yes
media	NUMBER	Yes
ref_c_id	NUMBER	Yes
type	NUMBER	No
wf_id	NUMBER	Yes

Table 125: ma_web_forms

Column Name	Datatype	Null
campaign_id	NUMBER	Yes
doc_id	NUMBER	Yes
name	VARCHAR2	Yes
sec_cookie	NUMBER	No
sec_login	NUMBER	No
sec_login_required	NUMBER	No
sec_tracking	NUMBER	No
set_cookie	NUMBER	No
shortcut	VARCHAR2	No
wf_id	NUMBER	No

Table 126: mail_addrs

Column Name	Datatype	Null
addr_id	NUMBER	No
email_address	VARCHAR2	No
first_name	VARCHAR2	Yes
last_name	VARCHAR2	Yes
notes	VARCHAR2	Yes

Table 127: mail_groups

Column Name	Datatype	Null
id	NUMBER	No
id_type	NUMBER	No
mgroup_id	NUMBER	No
mgroup_type	NUMBER	No
seq	NUMBER	No

Table 128: mail_list2addr

Column Name	Datatype	Null
addr_id	NUMBER	No
list_id	NUMBER	No

Table 129: mail_lists

Column Name	Datatype	Null
list_id	NUMBER	No
notes	VARCHAR2	Yes

Table 130: mailboxes

Column Name	Datatype	Null
ar_filter_type	NUMBER	No

Table 130: mailboxes (Continued)

Column Name	Datatype	Null
cert	VARCHAR2	Yes
default_mbox	NUMBER	No
desc_size	NUMBER	No
discard_addrs	VARCHAR2	Yes
discard_body	VARCHAR2	Yes
discard_files	VARCHAR2	Yes
discard_hdr	VARCHAR2	Yes
discard_subj	VARCHAR2	Yes
discard_types	VARCHAR2	Yes
display_name	VARCHAR2	Yes
enabled	NUMBER	No
fa_enabled	NUMBER	No
fa_size	NUMBER	No
force_reply_between	NUMBER	No
from_address	VARCHAR2	Yes
interface_id	NUMBER	No
key_passwd	VARCHAR2	Yes
mailbox_id	NUMBER	No
name	VARCHAR2	No
pop_account	VARCHAR2	Yes
pop_passwd	VARCHAR2	Yes
pop_server	VARCHAR2	Yes
priv_key	VARCHAR2	Yes

Table 130: mailboxes (Continued)

Column Name	Datatype	Null
pull_limit	NUMBER	No
reject_addr	VARCHAR2	Yes
reply_to	VARCHAR2	Yes
sa_max_suggestions	NUMBER	No
save_bulk_msgs	NUMBER	No
save_returned_msgs	NUMBER	No
seq	NUMBER	Yes
smime_trust	NUMBER	Yes
ssl_method	NUMBER	Yes
ssl_trust	NUMBER	Yes
vis	NUMBER	Yes

Table 131: map2meta_ans

Column Name	Datatype	Null
lvl1_id	NUMBER	Yes
lvl2_id	NUMBER	Yes
lvl3_id	NUMBER	Yes
lvl4_id	NUMBER	Yes
lvl5_id	NUMBER	Yes
lvl6_id	NUMBER	Yes
m_id	NUMBER	No
tbl	NUMBER	No

Table 132: map_cust

Column Name	Datatype	Null
external_id	NUMBER	No
external_source	NUMBER	No
preferred	NUMBER	Yes
rn_id	NUMBER	Yes

Table 133: map_prod

Column Name	Datatype	Null
external_id	NUMBER	No
external_source	NUMBER	No
preferred	NUMBER	Yes
rn_id	NUMBER	Yes

Table 134: menu_items

Column Name	Datatype	Null
cf_id	NUMBER	No
id	NUMBER	No
seq	NUMBER	No
tbl	NUMBER	No

Table 135: meta_ans_vis

Column Name	Datatype	Null
cat_lvl1	NUMBER	No
cat_lvl2	NUMBER	Yes
cat_lvl3	NUMBER	Yes
cat_lvl4	NUMBER	Yes
cat_lvl5	NUMBER	Yes
cat_lvl6	NUMBER	Yes
lvl1_id	NUMBER	Yes
lvl2_id	NUMBER	Yes
lvl3_id	NUMBER	Yes
lvl4_id	NUMBER	Yes
lvl5_id	NUMBER	Yes
lvl6_id	NUMBER	Yes
m_id	NUMBER	No
prod_id	NUMBER	No

Table 136: meta_ans_vis

Column Name	Datatype	Null
admin_vis	NUMBER	Yes
enduser_vis	NUMBER	Yes
interface_id	NUMBER	No
lvl1_id	NUMBER	Yes

Table 136: meta_ans_vis (Continued)

Column Name	Datatype	Null
lvl2_id	NUMBER	Yes
lvl3_id	NUMBER	Yes
lvl4_id	NUMBER	Yes
lvl5_id	NUMBER	Yes
lvl6_id	NUMBER	Yes
m_id	NUMBER	No
tbl	NUMBER	No

Table 137: meta_answers

Column Name	Datatype	Null
m_id	NUMBER	No
notes	VARCHAR2	Yes
orig_ref_no	VARCHAR2	Yes
summary	VARCHAR2	Yes

Table 138: meta_map

Column Name	Datatype	Null
display	VARCHAR2	Yes
in_transform	VARCHAR2	Yes
map_table	VARCHAR2	Yes
namespace	VARCHAR2	No
out_transform	VARCHAR2	Yes

Table 138: meta_map (Continued)

Column Name	Datatype	Null
url	VARCHAR2	Yes
xml_tag	VARCHAR2	No

Table 139: msg_types

Column Name	Datatype	Null
enabled	NUMBER	No
interface_id	NUMBER	No
msg_type_id	NUMBER	No
send_html	NUMBER	No

Table 140: obj_access

Column Name	Datatype	Null
id	NUMBER	Yes
obj_id	NUMBER	Yes
obj_subtype	NUMBER	Yes
obj_type	NUMBER	Yes
perms	NUMBER	Yes
tbl	NUMBER	No

Table 141: offer_phrases

Column Name	Datatype	Null
freq_description	NUMBER	No
freq_keyword	NUMBER	No
freq_name	NUMBER	No
id	NUMBER	No
lang_id	NUMBER	No
tbl	NUMBER	No
word	VARCHAR2	No
word_count	NUMBER	No

Table 142: offer_trans

Column Name	Datatype	Null
acct_id	NUMBER	Yes
c_id	NUMBER	Yes
channel	NUMBER	No
created	DATE	Yes
i_id	NUMBER	Yes
interface_id	NUMBER	No
offer_id	NUMBER	Yes
offer_type	NUMBER	No
product_id	NUMBER	Yes
response	NUMBER	Yes

Table 142: offer_trans (Continued)

Column Name	Datatype	Null
target_r_name	VARCHAR2	Yes

Table 143: offers

Column Name	Datatype	Null
activated	NUMBER	No
channels	NUMBER	No
cnt	NUMBER	No
created	DATE	Yes
created_by	NUMBER	No
description	VARCHAR2	Yes
end_date	DATE	Yes
folder_id	NUMBER	Yes
guide	VARCHAR2	Yes
interest_opp_create	NUMBER	No
interest_url	VARCHAR2	Yes
interface_id	NUMBER	No
keywords	VARCHAR2	Yes
name	VARCHAR2	Yes
notes	VARCHAR2	Yes
offer_id	NUMBER	No
priority	NUMBER	Yes
product_id	NUMBER	Yes
seq	NUMBER	Yes

Table 143: offers (Continued)

Column Name	Datatype	Null
start_date	DATE	Yes
updated	DATE	Yes
updated_by	NUMBER	No
yes_cnt	NUMBER	No
yes_opp_create	NUMBER	No
yes_url	VARCHAR2	Yes

Table 144: org_addr_types

Column Name	Datatype	Null
oat_id	NUMBER	Yes
seq	NUMBER	Yes

Table 145: org_addrs

Column Name	Datatype	Null
city	VARCHAR2	Yes
country_id	NUMBER	Yes
oat_id	NUMBER	Yes
org_id	NUMBER	Yes
postal_code	VARCHAR2	Yes
prov_id	NUMBER	Yes
street	VARCHAR2	Yes

Table 146: orgs

Column Name	Datatype	Null
alt_name	VARCHAR	Yes
created	DATE	Yes
css_state	NUMBER	No
login	VARCHAR2	Yes
lvl10_id	NUMBER	Yes
lvl11_id	NUMBER	Yes
lvl12_id	NUMBER	Yes
lvl1_id	NUMBER	Yes
lvl2_id	NUMBER	Yes
lvl3_id	NUMBER	Yes
lvl4_id	NUMBER	Yes
lvl5_id	NUMBER	Yes
lvl6_id	NUMBER	Yes
lvl7_id	NUMBER	Yes
lvl8_id	NUMBER	Yes
lvl9_id	NUMBER	Yes
ma_state	NUMBER	No
name	VARCHAR2	No
org_id	NUMBER	No
password	VARCHAR2	Yes
rule_state	NUMBER	Yes
sa_state	NUMBER	No

Table 146: orgs (Continued)

Column Name	Datatype	Null
salesperson	NUMBER	Yes
source	NUMBER	Yes
updated	DATE	Yes

Table 147: phrases

Column Name	Datatype	Null
dormant	NUMBER	No
i_id	NUMBER	No
natt	NUMBER	No
ncat	NUMBER	No
ncust	NUMBER	No
nflds	NUMBER	No
nlive	NUMBER	No
nprod	NUMBER	No
nstaf	NUMBER	No
nsubj	NUMBER	No
word	VARCHAR2	No

Table 148: prod_links

Column Name	Datatype	Null
prod_id	NUMBER	No
lvl1_id	NUMBER	No

Table 148: prod_links (Continued)

Column Name	Datatype	Null
lvl2_id	NUMBER	Yes
lvl3_id	NUMBER	Yes
lvl4_id	NUMBER	Yes
lvl5_id	NUMBER	Yes
lvl6_id	NUMBER	Yes
tbl	NUMBER	No

Table 149: prodcat_notif

Column Name	Datatype	Null
c_id	NUMBER	No
interface_id	NUMBER	Yes
lvl1_id	NUMBER	No
lvl2_id	NUMBER	Yes
lvl3_id	NUMBER	Yes
lvl4_id	NUMBER	Yes
lvl5_id	NUMBER	Yes
lvl6_id	NUMBER	Yes
start_time	DATE	Yes
tbl	NUMBER	No

Table 150: profile2acd_mode

Column Name	Datatype	Null
aam_id	NUMBER	No
profile_id	NUMBER	No

Table 151: profile2intf

Column Name	Datatype	Null
interface_id	NUMBER	No
profile_id	NUMBER	No

Table 152: profile2queue

Column Name	Datatype	Null
profile_id	NUMBER	Yes
queue_id	NUMBER	Yes
seq	NUMBER	Yes

Table 153: profiles

Column Name	Datatype	Null
after_call	NUMBER	Yes
after_call_interval	NUMBER	Yes
ani_filter	VARCHAR2	Yes

Table 153: profiles (Continued)

Column Name	Datatype	Null
appdata_filter	VARCHAR2	Yes
console	NUMBER	Yes
contact_perms	NUMBER	No
css_perms	NUMBER	No
cti_interface_id	NUMBER	Yes
custom_func	VARCHAR2	Yes
custom_url	VARCHAR2	Yes
dnis_filter	VARCHAR2	Yes
dtmf_filter	VARCHAR2	Yes
global_perms	NUMBER	No
lsk_policy	NUMBER	Yes
ma_perms	NUMBER	No
notes	VARCHAR2	Yes
org_perms	NUMBER	No
popup_type	NUMBER	Yes
profile_id	NUMBER	No
sa_perms	NUMBER	No
seq	NUMBER	Yes
sk_display	NUMBER	Yes
sk_max_qty	NUMBER	Yes
sk_policy	NUMBER	Yes
sk_pull_qty	NUMBER	Yes
tabset_data	NUMBER	Yes

Table 153: profiles (Continued)

Column Name	Datatype	Null
view_id	NUMBER	Yes

Table 154: provinces

Column Name	Datatype	Null
country_id	NUMBER	Yes
prov_id	NUMBER	No

Table 155: queue_stats

Column Name	Datatype	Null
created	DATE	Yes
i_id	NUMBER	No
queue_action	NUMBER	Yes
queue_id	NUMBER	Yes

Table 156: queues

Column Name	Datatype	Null
crit_lvl	NUMBER	Yes
default_queue	NUMBER	No
id	NUMBER	No
qtype	NUMBER	Yes
rrobin_enabled	NUMBER	No

Table 156: queues (Continued)

Column Name	Datatype	Null
rrobin_last_id	NUMBER	Yes
seq	NUMBER	Yes
warn_lvl	NUMBER	Yes

Table 157: response_reqs

Column Name	Datatype	Null
interface_id	NUMBER	Yes
resp_within	NUMBER	Yes
rr_id	NUMBER	No
rr_set	NUMBER	Yes
rr_type	NUMBER	Yes
rslv_within	NUMBER	Yes
sla_id	NUMBER	Yes

Table 158: rnl_chat_activities

Column Name	Datatype	Null
acct_id	NUMBER	Yes
agent_session_number	VARCHAR2	Yes
id1	NUMBER	Yes
rnl_chat_id	NUMBER	Yes
rnl_queue_id	NUMBER	Yes
timecreated	DATE	Yes

Table 158: rnl_chat_activities (Continued)

Column Name	Datatype	Null
trancode	NUMBER	Yes
url_pushed	VARCHAR2	Yes

Table 159: rnl_chat2ma

Column Name	Datatype	Null
campaign_id	NUMBER	Yes
mailing_id	NUMBER	Yes
rnl_chat_id	NUMBER	Yes

Table 160: rnl_chats

Column Name	Datatype	Null
agent_id	NUMBER	No
agent_session_number	VARCHAR2	Yes
applet_handle_time	NUMBER	Yes
c_id	NUMBER	Yes
call_type	NUMBER	Yes
cat_lvl1	NUMBER	Yes
cat_lvl2	NUMBER	Yes
cat_lvl3	NUMBER	Yes
cat_lvl4	NUMBER	Yes
cat_lvl5	NUMBER	Yes
cat_lvl6	NUMBER	Yes

Table 160: rnl_chats (Continued)

Column Name	Datatype	Null
client_browser	VARCHAR2	Yes
client_ipaddr	VARCHAR2	Yes
client_os	VARCHAR2	Yes
email	VARCHAR2	Yes
first_name	VARCHAR2	Yes
i_id	NUMBER	Yes
interface_id	NUMBER	Yes
last_name	VARCHAR2	Yes
num_responses	NUMBER	Yes
prod_lvl1	NUMBER	Yes
prod_lvl2	NUMBER	Yes
prod_lvl3	NUMBER	Yes
prod_lvl4	NUMBER	Yes
prod_lvl5	NUMBER	Yes
prod_lvl6	NUMBER	Yes
rnl_chat_id	NUMBER	No
rnl_chat_sourc	NUMBER	Yes
rnl_queue_id	NUMBER	Yes
session_num	NUMBER	No
session_type	NUMBER	No
status	NUMBER	No
termination_code	NUMBER	Yes
time_end	DATE	Yes

Table 160: rnl_chats (Continued)

Column Name	Datatype	Null
time_request	DATE	Yes
time_start	DATE	Yes
total_resp_time	NUMBER	Yes
user_session_number	VARCHAR2	Yes

Table 161: rnl_staff_activity

Column Name	Datatype	Null
acct_id	NUMBER	Yes
staff_login_id	NUMBER	Yes
timestamp	DATE	Yes
tran_code	NUMBER	Yes

Table 162: rnl_staff_engage

Column Name	Datatype	Null
acct_id	NUMBER	No
end_eng	DATE	Yes
staff_login_id	NUMBER	No
start_eng	DATE	Yes

Table 163: rnl_staff_login

Column Name	Datatype	Null
acct_id	NUMBER	No
agent_ipaddr	VARCHAR2	Yes
call_back_requests	NUMBER	Yes
chat_complete_count	NUMBER	Yes
chats_transferred	NUMBER	Yes
conferences	NUMBER	Yes
decl_conferences	NUMBER	Yes
decl_transfers	NUMBER	Yes
declined_count	NUMBER	No
expired_count	NUMBER	No
interface_id	NUMBER	Yes
staff_login_id	NUMBER	No
time_engaged	NUMBER	Yes
time_login	DATE	Yes
time_logoff	DATE	Yes
transfers	NUMBER	Yes

Table 164: rr2holidays

Column Name	Datatype	Null
holiday_id	NUMBER	Yes
rr_id	NUMBER	Yes

Table 165: rr_intervals

Column Name	Datatype	Null
day	NUMBER	Yes
rr_id	NUMBER	Yes
tm_end	DATE	Yes
tm_start	DATE	Yes

Table 166: rule_alerts

Column Name	Datatype	Null
alert_action	NUMBER	No
alert_time	DATE	Yes
entity_id	NUMBER	No
escl_lvl	NUMBER	No
id	NUMBER	No
revalidate	NUMBER	No
rule_id	NUMBER	No
rule_type	NUMBER	No
ruleact_seq	NUMBER	No

Table 167: rule_escalations

Column Name	Datatype	Null
escl_level	VARCHAR2	No

Table 167: rule_escalations (Continued)

Column Name	Datatype	Null
id	NUMBER	No
rb_status	NUMBER	Yes
rule_key	NUMBER	Yes
rule_type	NUMBER	Yes

Table 168: rule_log

Column Name	Datatype	Null
match_dttm	DATE	Yes
rec_id	NUMBER	Yes
rule_name	VARCHAR2	Yes
rule_type	NUMBER	Yes
seq	NUMBER	Yes

Table 169: rule_state_xitions

Column Name	Datatype	Null
archive_id	NUMBER	No
from_state	NUMBER	No
to_state	NUMBER	No

Table 170: rule_states

Column Name	Datatype	Null
archive_id	NUMBER	No
func_ind	NUMBER	No
id	NUMBER	No
initial_state	NUMBER	Yes
name	VARCHAR2	No
notes	VARCHAR2	Yes
rb_status	NUMBER	Yes
rs_key	NUMBER	No
rule_type	NUMBER	No
xml_block	VARCHAR2	Yes

Table 171: ruleacts

Column Name	Datatype	Null
action	NUMBER	No
arg_id	NUMBER	Yes
argument	VARCHAR2	Yes
date_arg	DATE	Yes
dttm_arg	TIME	Yes
rule_key	NUMBER	No
seq	NUMBER	No

Table 172: ruleconds

Column Name	Datatype	Null
date_val	DATE	Yes
dttm_val	TIME	
field_id	NUMBER	No
field_src	NUMBER	No
filter_id	NUMBER	No
id	NUMBER	No
oper	NUMBER	No
rule_key	NUMBER	No
seq	NUMBER	No
subseq	NUMBER	No
val	VARCHAR2	Yes

Table 173: rules

Column Name	Datatype	Null
cond_expr	VARCHAR2	Yes
enabled	NUMBER	No
id	NUMBER	No
name	VARCHAR2	No
notes	VARCHAR2	Yes
r_key	NUMBER	No
seq	NUMBER	No

Table 173: rules (Continued)

Column Name	Datatype	Null
state_key	NUMBER	No

Table 174: rules_archive

Column Name	Datatype	Null
activate_date	DATE	Yes
archive_date	DATE	Yes
create_date	DATE	Yes
created_by	NUMBER	Yes
foreign_id	NUMBER	Yes
id	NUMBER	No
last_modified	DATE	Yes
modified_by	NUMBER	Yes
rb_path	VARCHAR2	Yes
rb_status	NUMBER	No
rule_type	NUMBER	No
version	VARCHAR2	Yes

Table 175: rx_email

Column Name	Datatype	Null
created	NUMBER	No
email	VARCHAR2	No
mailbox_id	NUMBER	No

Table 176: sa_contact_roles

Column Name	Datatype	Null
cr_id	NUMBER	No
notes	VARCHAR2	Yes
seq	NUMBER	No

Table 177: sa_opp2contacts

Column Name	Datatype	Null
c_id	NUMBER	No
cr_id	NUMBER	Yes
oc_primary	NUMBER	No
op_id	NUMBER	No

Table 178: sa_opp_sources

Column Name	Datatype	Null
name	VARCHAR2	No
notes	VARCHAR2	Yes
os_id	NUMBER	No
seq	NUMBER	No

Table 179: sa_opportunities

Column Name	Datatype	Null
acct_lvl10_id	NUMBER	Yes
acct_lvl11_id	NUMBER	Yes
acct_lvl12_id	NUMBER	Yes
acct_lvl1_id	NUMBER	Yes
acct_lvl2_id	NUMBER	Yes
acct_lvl3_id	NUMBER	Yes
acct_lvl4_id	NUMBER	Yes
acct_lvl5_id	NUMBER	Yes
acct_lvl6_id	NUMBER	Yes
acct_lvl7_id	NUMBER	Yes
acct_lvl8_id	NUMBER	Yes
acct_lvl9_id	NUMBER	Yes
assgn_acct_id	NUMBER	Yes
assgn_group_id	NUMBER	Yes
c_id	NUMBER	No
campaign_id	NUMBER	Yes
closed	DATE	Yes
closed_value	NUMBER	Yes
closed_value_curr_id	NUMBER	Yes
closed_value_rate_id	NUMBER	Yes
created	DATE	Yes
created_by	NUMBER	Yes

Table 179: sa_opportunities (Continued)

Column Name	Datatype	Null
dormant	NUMBER	No
escldate	DATE	Yes
escllevel	NUMBER	Yes
forecast_close	DATE	Yes
initial_contact	DATE	Yes
interface_id	NUMBER	No
mgr_commit	NUMBER	No
mgr_value	NUMBER	Yes
mgr_value_curr_id	NUMBER	Yes
mgr_value_rate_id	NUMBER	Yes
name	VARCHAR2	Yes
op_id	NUMBER	No
org_id	NUMBER	Yes
recall	DATE	Yes
rep_commit	NUMBER	No
rep_value	NUMBER	Yes
rep_value_curr_id	NUMBER	Yes
rep_value_rate_id	NUMBER	Yes
rule_state	NUMBER	Yes
source	NUMBER	No
stage_id	NUMBER	Yes
status_id	NUMBER	Yes
status_type	NUMBER	Yes

Table 179: sa_opportunities (Continued)

Column Name	Datatype	Null
strategy_id	NUMBER	Yes
summary	VARCHAR2	Yes
terr_id	NUMBER	Yes
terr_lvl10_id	NUMBER	Yes
terr_lvl11_id	NUMBER	Yes
terr_lvl12_id	NUMBER	Yes
terr_lvl1_id	NUMBER	Yes
terr_lvl2_id	NUMBER	Yes
terr_lvl3_id	NUMBER	Yes
terr_lvl4_id	NUMBER	Yes
terr_lvl5_id	NUMBER	Yes
terr_lvl6_id	NUMBER	Yes
terr_lvl7_id	NUMBER	Yes
terr_lvl8_id	NUMBER	Yes
terr_lvl9_id	NUMBER	Yes
updated	DATE	Yes
updated_by	NUMBER	Yes

Table 180: sa_period2acct

Column Name	Datatype	Null
acct_id	NUMBER	No
amount	NUMBER	No
amount_curr_id	NUMBER	No

Table 180: sa_period2acct (Continued)

Column Name	Datatype	Null
amount_rate_id	NUMBER	Yes
sp_id	NUMBER	No

Table 181: sa_price_schedules

Column Name	Datatype	Null
currency_id	NUMBER	No
disabled	NUMBER	No
notes	VARCHAR2	Yes
schedule_id	NUMBER	No
seq	NUMBER	No

Table 182: sa_prod2quotes

Column Name	Datatype	Null
adjusted_desc	VARCHAR2	Yes
adjusted_id	VARCHAR2	Yes
adjusted_name	VARCHAR2	Yes
adjusted_price	NUMBER	Yes
adjusted_price_curr_id	NUMBER	Yes
adjusted_price_rate_id	NUMBER	Yes
adjusted_total	NUMBER	Yes
adjusted_total_curr_id	NUMBER	Yes
adjusted_total_rate_id	NUMBER	Yes

Table 182: sa_prod2quotes (Continued)

Column Name	Datatype	Null
discount	NUMBER	No
notes	VARCHAR2	Yes
original_desc	VARCHAR2	Yes
original_id	VARCHAR2	Yes
original_name	VARCHAR2	Yes
original_price	NUMBER	Yes
original_price_curr_id	NUMBER	Yes
original_price_rate_id	NUMBER	Yes
product_id	NUMBER	Yes
qty	NUMBER	No
quote_id	NUMBER	No
seq	NUMBER	No

Table 183: sa_prod2sched

Column Name	Datatype	Null
notes	VARCHAR2	Yes
price	NUMBER	No
price_curr_id	NUMBER	No
price_rate_id	NUMBER	Yes
product_id	NUMBER	No
schedule_end	DATE	Yes
schedule_id	NUMBER	No
schedule_start	DATE	Yes

Table 184: sa_products

Column Name	Datatype	Null
cnt	NUMBER	No
cua_exclude	NUMBER	Yes
disabled	NUMBER	No
folder_id	NUMBER	Yes
id	VARCHAR2	Yes
product_id	NUMBER	No
seq	NUMBER	No
updated	DATE	No
yes_cnt	NUMBER	No

Table 185: sa_purchased_products

Column Name	Datatype	Null
c_id	NUMBER	Yes
cua_c_id	NUMBER	Yes
finalized_by	NUMBER	Yes
license_end	DATE	Yes
license_start	DATE	Yes
notes	VARCHAR2	Yes
op_id	NUMBER	Yes
org_id	NUMBER	Yes
price	NUMBER	Yes

Table 185: sa_purchased_products (Continued)

Column Name	Datatype	Null
price_curr_id	NUMBER	Yes
price_rate_id	NUMBER	Yes
product_id	NUMBER	Yes
purchase_date	DATE	Yes
quote_id	NUMBER	Yes
serial_number	VARCHAR2	Yes

Table 186: sa_quote_templates

Column Name	Datatype	Null
disabled	NUMBER	No
folder_id	NUMBER	Yes
notes	VARCHAR2	Yes
seq	NUMBER	No
template_id	NUMBER	No

Table 187: sa_quotes

Column Name	Datatype	Null
adj_total	NUMBER	Yes
adj_total_curr_id	NUMBER	Yes
adj_total_rate_id	NUMBER	Yes
created	DATE	Yes
created_by	NUMBER	No

Table 187: sa_quotes (Continued)

Column Name	Datatype	Null
discount	NUMBER	No
forecast	NUMBER	No
name	VARCHAR2	No
notes	VARCHAR2	Yes
offer_end	DATE	Yes
offer_start	DATE	Yes
op_id	NUMBER	Yes
quote_id	NUMBER	No
schedule_id	NUMBER	Yes
sent	DATE	Yes
sent_to	VARCHAR2	Yes
status	NUMBER	Yes
template_id	NUMBER	Yes
total	NUMBER	Yes
total_curr_id	NUMBER	Yes
total_rate_id	NUMBER	Yes
updated	DATE	Yes
updated_by	NUMBER	No

Table 188: sa_sales_periods

Column Name	Datatype	Null
interface_id	NUMBER	No
notes	VARCHAR2	Yes

Table 188: sa_sales_periods (Continued)

Column Name	Datatype	Null
sp_end	DATE	Yes
sp_id	NUMBER	No
sp_start	DATE	Yes

Table 189: sa_stages

Column Name	Datatype	Null
forecast	NUMBER	No
forecast_pct	NUMBER	Yes
notes	VARCHAR2	Yes
seq	NUMBER	No
stage_id	NUMBER	No
strategy_id	NUMBER	No

Table 190: sa_strategies

Column Name	Datatype	Null
disabled	NUMBER	No
notes	VARCHAR2	Yes
seq	NUMBER	No
strategy_id	NUMBER	No

Table 191: sa_task_instances

Column Name	Datatype	Null
assgn_acct_id	NUMBER	Yes
c_id	NUMBER	Yes
completed	DATE	Yes
created	DATE	Yes
due_date	DATE	Yes
name	VARCHAR2	No
notes	VARCHAR2	Yes
op_id	NUMBER	Yes
org_id	NUMBER	Yes
planned_completion	DATE	Yes
source	NUMBER	No
task_id	NUMBER	Yes
ti_id	NUMBER	No
updated	DATE	Yes

Table 192: sa_tasks

Column Name	Datatype	Null
notes	VARCHAR2	Yes
outlook	NUMBER	No
owner	NUMBER	Yes
seq	NUMBER	No

Table 192: sa_tasks (Continued)

Column Name	Datatype	Null
stage_id	NUMBER	No
task_id	NUMBER	No
task_interval	NUMBER	Yes

Table 193: sa_territories

Column Name	Datatype	Null
disabled	NUMBER	No
lvl10_id	NUMBER	Yes
lvl11_id	NUMBER	Yes
lvl12_id	NUMBER	Yes
lvl1_id	NUMBER	Yes
lvl2_id	NUMBER	Yes
lvl3_id	NUMBER	Yes
lvl4_id	NUMBER	Yes
lvl5_id	NUMBER	Yes
lvl6_id	NUMBER	Yes
lvl7_id	NUMBER	Yes
lvl8_id	NUMBER	Yes
lvl9_id	NUMBER	Yes
notes	VARCHAR2	Yes
seq	NUMBER	No
terr_id	NUMBER	No

Table 194: sec_contacts

Column Name	Datatype	Null
c_id	NUMBER	No
id	NUMBER	No

Table 195: segment_attributes

Column Name	Datatype	Null
attr_id	NUMBER	No
attr_type	NUMBER	No
col_name	VARCHAR2	No
default_weight	NUMBER	Yes
id	NUMBER	Yes
name	VARCHAR2	No
tbl_name	VARCHAR2	No
weight	NUMBER	Yes

Table 196: segments

Column Name	Datatype	Null
attr_id	NUMBER	No
interface_id	NUMBER	No
seg_id	NUMBER	No
weight	NUMBER	Yes

Table 197: session_summary

Column Name	Datatype	Null
ep_adp	NUMBER	Yes
ep_alp	NUMBER	Yes
ep_ask	NUMBER	Yes
ep_cls_adp	NUMBER	Yes
ep_cls_alp	NUMBER	Yes
ep_g_adp	NUMBER	Yes
ep_g_alp	NUMBER	Yes
ep_other	NUMBER	Yes
ep_rel_adp	NUMBER	Yes
ep_sh	NUMBER	Yes
excl_sessions	NUMBER	Yes
interface_id	NUMBER	No
max_pcnt	NUMBER	Yes
max_time	NUMBER	Yes
min_pcnt	NUMBER	Yes
min_time	NUMBER	Yes
npa_adp	NUMBER	Yes
npa_alp	NUMBER	Yes
npa_ask	NUMBER	Yes
npa_cls_adp	NUMBER	Yes
npa_cls_alp	NUMBER	Yes
npa_g_adp	NUMBER	Yes

Table 197: session_summary (Continued)

Column Name	Datatype	Null
npa_g_alp	NUMBER	Yes
npa_other	NUMBER	Yes
npa_rel_adp	NUMBER	Yes
npa_sh	NUMBER	Yes
pa_adp	NUMBER	Yes
pa_alp	NUMBER	Yes
pa_ask	NUMBER	Yes
pa_cls_adp	NUMBER	Yes
pa_cls_alp	NUMBER	Yes
pa_g_adp	NUMBER	Yes
pa_g_alp	NUMBER	Yes
pa_other	NUMBER	Yes
pa_rel_adp	NUMBER	Yes
pa_sh	NUMBER	Yes
sc_ans	NUMBER	Yes
sc_ans_conf	NUMBER	Yes
sc_ans_srch_conf	NUMBER	Yes
sc_ans_srch_sub	NUMBER	Yes
sc_ans_sub	NUMBER	Yes
sc_none_conf	NUMBER	Yes
sc_none_sub	NUMBER	Yes
sc_pa	NUMBER	Yes
sc_search	NUMBER	Yes

Table 197: session_summary (Continued)

Column Name	Datatype	Null
sc_srch_conf	NUMBER	Yes
sc_srch_sub	NUMBER	Yes
sp_adp	NUMBER	Yes
sp_alp	NUMBER	Yes
sp_ask	NUMBER	Yes
sp_cls_adp	NUMBER	Yes
sp_cls_alp	NUMBER	Yes
sp_g_adp	NUMBER	Yes
sp_g_alp	NUMBER	Yes
sp_other	NUMBER	Yes
sp_rel_adp	NUMBER	Yes
sp_sh	NUMBER	Yes
stat_date	DATE	Yes
tot_pcnt	NUMBER	Yes
tot_sessions	NUMBER	Yes
tot_date	NUMBER	Yes
tr_adp2adp	NUMBER	Yes
tr_adp2alp	NUMBER	Yes
tr_adp2ask	NUMBER	Yes
tr_adp2cls_adp	NUMBER	Yes
tr_adp2cls_alp	NUMBER	Yes
tr_adp2g_adp	NUMBER	Yes
tr_adp2g_alp	NUMBER	Yes

Table 197: session_summary (Continued)

Column Name	Datatype	Null
tr_adp2rel_adp	NUMBER	Yes
tr_adp2sh	NUMBER	Yes
tr_alp2adp	NUMBER	Yes
tr_alp2alp	NUMBER	Yes
tr_alp2ask	NUMBER	Yes
tr_alp2cls_adp	NUMBER	Yes
tr_alp2cls_alp	NUMBER	Yes
tr_alp2g_adp	NUMBER	Yes
tr_alp2g_alp	NUMBER	Yes
tr_alp2rel_adp	NUMBER	Yes
tr_alp2sh	NUMBER	Yes
tr_ask2adp	NUMBER	Yes
tr_ask2alp	NUMBER	Yes
tr_ask2ask	NUMBER	Yes
tr_ask2cls_adp	NUMBER	Yes
tr_ask2cls_alp	NUMBER	Yes
tr_ask2g_adp	NUMBER	Yes
tr_ask2g_alp	NUMBER	Yes
tr_ask2rel_adp	NUMBER	Yes
tr_ask2sh	NUMBER	Yes
tr_cls_adp2adp	NUMBER	Yes
tr_cls_adp2alp	NUMBER	Yes
tr_cls_adp2ask	NUMBER	Yes

Table 197: session_summary (Continued)

Column Name	Datatype	Null
tr_cls_adp2cls_adp	NUMBER	Yes
tr_cls_adp2cls_alp	NUMBER	Yes
tr_cls_adp2g_adp	NUMBER	Yes
tr_cls_adp2g_alp	NUMBER	Yes
tr_cls_adp2rel_adp	NUMBER	Yes
tr_cls_adp2sh	NUMBER	Yes
tr_cls_alp2adp	NUMBER	Yes
tr_cls_alp2alp	NUMBER	Yes
tr_cls_alp2ask	NUMBER	Yes
tr_cls_alp2cls_adp	NUMBER	Yes
tr_cls_alp2cls_alp	NUMBER	Yes
tr_cls_alp2g_adp	NUMBER	Yes
tr_cls_alp2g_alp	NUMBER	Yes
tr_cls_alp2rel_adp	NUMBER	Yes
tr_cls_alp2sh	NUMBER	Yes
tr_g_adp2adp	NUMBER	Yes
tr_g_adp2alp	NUMBER	Yes
tr_g_adp2ask	NUMBER	Yes
tr_g_adp2cls_adp	NUMBER	Yes
tr_g_adp2cls_alp	NUMBER	Yes
tr_g_adp2g_adp	NUMBER	Yes
tr_g_adp2g_alp	NUMBER	Yes
tr_g_adp2rel_adp	NUMBER	Yes

Table 197: session_summary (Continued)

Column Name	Datatype	Null
tr_g_adp2sh	NUMBER	Yes
tr_g_alp2adp	NUMBER	Yes
tr_g_alp2alp	NUMBER	Yes
tr_g_alp2ask	NUMBER	Yes
tr_g_alp2cls_adp	NUMBER	Yes
tr_g_alp2cls_alp	NUMBER	Yes
tr_g_alp2g_adp	NUMBER	Yes
tr_g_alp2g_alp	NUMBER	Yes
tr_g_alp2rel_adp	NUMBER	Yes
tr_g_alp2sh	NUMBER	Yes
tr_rel_adp2adp	NUMBER	Yes
tr_rel_adp2alp	NUMBER	Yes
tr_rel_adp2ask	NUMBER	Yes
tr_rel_adp2cls_adp	NUMBER	Yes
tr_rel_adp2cls_alp	NUMBER	Yes
tr_rel_adp2g_adp	NUMBER	Yes
tr_rel_adp2g_alp	NUMBER	Yes
tr_rel_adp2rel_adp	NUMBER	Yes
tr_rel_adp2sh	NUMBER	Yes
tr_sh2adp	NUMBER	Yes
tr_sh2alp	NUMBER	Yes
tr_sh2ask	NUMBER	Yes
tr_sh2cls_adp	NUMBER	Yes

Table 197: session_summary (Continued)

Column Name	Datatype	Null
tr_sh2cls_alp	NUMBER	Yes
tr_sh2g_adp	NUMBER	Yes
tr_sh2g_alp	NUMBER	Yes
tr_sh2rel_adp	NUMBER	Yes
tr_sh2sh	NUMBER	Yes
track_adp	NUMBER	Yes
track_alp	NUMBER	Yes
track_ask	NUMBER	Yes
track_cls_adp	NUMBER	Yes
track_cls_alp	NUMBER	Yes
track_g_adp	NUMBER	Yes
track_g_alp	NUMBER	Yes
track_other	NUMBER	Yes
track_rel_adp	NUMBER	Yes
track_sh	NUMBER	Yes
tt_adp2adp	NUMBER	Yes
tt_adp2alp	NUMBER	Yes
tt_adp2ask	NUMBER	Yes
tt_adp2cls_adp	NUMBER	Yes
tt_adp2cls_alp	NUMBER	Yes
tt_adp2g_adp	NUMBER	Yes
tt_adp2g_alp	NUMBER	Yes
tt_adp2rel_adp	NUMBER	Yes

Table 197: session_summary (Continued)

Column Name	Datatype	Null
tt_adp2sh	NUMBER	Yes
tt_alp2adp	NUMBER	Yes
tt_alp2alp	NUMBER	Yes
tt_alp2ask	NUMBER	Yes
tt_alp2cls_adp	NUMBER	Yes
tt_alp2cls_alp	NUMBER	Yes
tt_alp2g_adp	NUMBER	Yes
tt_alp2g_alp	NUMBER	Yes
tt_alp2rel_adp	NUMBER	Yes
tt_alp2sh	NUMBER	Yes
tt_ask2adp	NUMBER	Yes
tt_ask2alp	NUMBER	Yes
tt_ask2ask	NUMBER	Yes
tt_ask2cls_adp	NUMBER	Yes
tt_ask2cls_alp	NUMBER	Yes
tt_ask2g_adp	NUMBER	Yes
tt_ask2g_alp	NUMBER	Yes
tt_ask2rel_adp	NUMBER	Yes
tt_ask2sh	NUMBER	Yes
tt_cls_adp2adp	NUMBER	Yes
tt_cls_adp2alp	NUMBER	Yes
tt_cls_adp2ask	NUMBER	Yes
tt_cls_adp2cls_adp	NUMBER	Yes

Table 197: session_summary (Continued)

Column Name	Datatype	Null
tt_cls_adp2cls_alp	NUMBER	Yes
tt_cls_adp2g_adp	NUMBER	Yes
tt_cls_adp2g_alp	NUMBER	Yes
tt_cls_adp2rel_adp	NUMBER	Yes
tt_cls_adp2sh	NUMBER	Yes
tt_cls_alp2adp	NUMBER	Yes
tt_cls_alp2alp	NUMBER	Yes
tt_cls_alp2ask	NUMBER	Yes
tt_cls_alp2cls_adp	NUMBER	Yes
tt_cls_alp2cls_alp	NUMBER	Yes
tt_cls_alp2g_adp	NUMBER	Yes
tt_cls_alp2g_alp	NUMBER	Yes
tt_cls_alp2rel_adp	NUMBER	Yes
tt_cls_alp2sh	NUMBER	Yes
tt_g_adp2adp	NUMBER	Yes
tt_g_adp2alp	NUMBER	Yes
tt_g_adp2ask	NUMBER	Yes
tt_g_adp2cls_adp	NUMBER	Yes
tt_g_adp2cls_alp	NUMBER	Yes
tt_g_adp2g_adp	NUMBER	Yes
tt_g_adp2g_alp	NUMBER	Yes
tt_g_adp2rel_adp	NUMBER	Yes
tt_g_adp2sh	NUMBER	Yes

Table 197: session_summary (Continued)

Column Name	Datatype	Null
tt_g_alp2adp	NUMBER	Yes
tt_g_alp2alp	NUMBER	Yes
tt_g_alp2ask	NUMBER	Yes
tt_g_alp2cls_adp	NUMBER	Yes
tt_g_alp2cls_alp	NUMBER	Yes
tt_g_alp2g_adp	NUMBER	Yes
tt_g_alp2g_alp	NUMBER	Yes
tt_g_alp2rel_adp	NUMBER	Yes
tt_g_alp2sh	NUMBER	Yes
tt_rel_adp2adp	NUMBER	Yes
tt_rel_adp2alp	NUMBER	Yes
tt_rel_adp2ask	NUMBER	Yes
tt_rel_adp2cls_adp	NUMBER	Yes
tt_rel_adp2cls_alp	NUMBER	Yes
tt_rel_adp2g_adp	NUMBER	Yes
tt_rel_adp2g_alp	NUMBER	Yes
tt_rel_adp2rel_adp	NUMBER	Yes
tt_rel_adp2sh	NUMBER	Yes
tt_sh2adp	NUMBER	Yes
tt_sh2alp	NUMBER	Yes
tt_sh2ask	NUMBER	Yes
tt_sh2cls_adp	NUMBER	Yes
tt_sh2cls_alp	NUMBER	Yes

Table 197: session_summary (Continued)

Column Name	Datatype	Null
tt_sh2g_adp	NUMBER	Yes
tt_sh2g_alp	NUMBER	Yes
tt_sh2rel_adp	NUMBER	Yes
tt_sh2sh	NUMBER	Yes

Table 198: similar_search_links

Column Name	Datatype	Null
access_time	DATE	Yes
active	NUMBER	Yes
from_id	NUMBER	No
ml_weight	NUMBER	Yes
to_id	NUMBER	No
user_weight	NUMBER	Yes

Table 199: similar_searches

Column Name	Datatype	Null
active	NUMBER	Yes
id	NUMBER	No
srch	VARCHAR2	Yes
stem	VARCHAR2	Yes

Table 200: sla2ans_access

Column Name	Datatype	Null
access_id	NUMBER	Yes
sla_id	NUMBER	Yes

Table 201: sla_instances

Column Name	Datatype	Null
activedate	DATE	Yes
expiredate	DATE	Yes
inc_chat	NUMBER	Yes
inc_csr	NUMBER	Yes
inc_email	NUMBER	Yes
inc_total	NUMBER	Yes
inc_web	NUMBER	Yes
owner_id	NUMBER	Yes
owner_tbl	NUMBER	Yes
sla_id	NUMBER	Yes
slai_id	NUMBER	No
sla_set	NUMBER	Yes
state	NUMBER	Yes
time_billed	NUMBER	Yes

Table 202: slas

Column Name	Datatype	Null
active	NUMBER	Yes
created	DATE	Yes
len_units	NUMBER	Yes
length	NUMBER	Yes
qty_chat	NUMBER	Yes
qty_csr	NUMBER	Yes
qty_email	NUMBER	Yes
qty_inc	NUMBER	Yes
qty_time	NUMBER	Yes
qty_web	NUMBER	Yes
seq	NUMBER	Yes
sla_id	NUMBER	No
sla_set	NUMBER	Yes
time_limit	NUMBER	Yes
web_access	NUMBER	Yes

Table 203: stats

Column Name	Datatype	Null
ans_viewed	NUMBER	No
api_incidents	NUMBER	No
assists	NUMBER	No

Table 203: stats (Continued)

Column Name	Datatype	Null
avg_resp_time	NUMBER	No
avg_soln_time	NUMBER	No
email_assists	NUMBER	No
guided_search	NUMBER	No
hits	NUMBER	No
incident_backlog	NUMBER	No
interface_id	NUMBER	Yes
kb_assists	NUMBER	No
ma_assists	NUMBER	No
new_incidents	NUMBER	No
resp_incidents	NUMBER	No
searches	NUMBER	No
sessions	NUMBER	No
solved_incidents	NUMBER	No
stat_date	DATE	Yes

Table 204: statuses

Column Name	Datatype	Null
seq	NUMBER	Yes
status_id	NUMBER	No
tbl	NUMBER	No
type_id	NUMBER	No

Table 205: std_content

Column Name	Datatype	Null
code	NUMBER	No
folder_id	NUMBER	Yes
html_val	CLOB	Yes
kbd_id	VARCHAR2	Yes
name	VARCHAR2	No
seq	NUMBER	Yes
tbl	NUMBER	No
type	NUMBER	No
val	CLOB	Yes

Table 206: target2offers

Column Name	Datatype	Null
offer_id	NUMBER	No
target_r_id	NUMBER	Yes

Table 207: threads

Column Name	Datatype	Null
acct_id	NUMBER	Yes
c_id	NUMBER	Yes
ei	NUMBER	Yes

Table 207: threads (Continued)

Column Name	Datatype	Null
entered	DATE	Yes
entry_type	NUMBER	No
id	NUMBER	No
mail_hdr	VARCHAR2	Yes
note	CLOB	No
tbl	NUMBER	No

Table 208: time_billed

Column Name	Datatype	Null
acct_id	NUMBER	Yes
bill_date	DATE	Yes
i_id	NUMBER	No
minutes	NUMBER	Yes
notes	VARCHAR2	Yes
task_id	NUMBER	Yes

Table 209: topic_word_phrases

Column Name	Datatype	Null
stem	VARCHAR2	No
tw_id	NUMBER	No
word	VARCHAR2	No

Table 210: topic_words

Column Name	Datatype	Null
a_id	NUMBER	Yes
interface_id	NUMBER	No
label	VARCHAR2	No
state	NUMBER	No
text	VARCHAR2	Yes
title	VARCHAR2	Yes
tw_id	NUMBER	No
tw_type	NUMBER	No
url	VARCHAR2	Yes

Table 211: transactions

Column Name	Datatype	Null
acct_id	NUMBER	Yes
created	DATE	Yes
description	VARCHAR2	Yes
id	NUMBER	No
id1	NUMBER	Yes
id2	NUMBER	Yes
id3	NUMBER	Yes
interface_id	NUMBER	Yes
tbl	NUMBER	No

Table 211: transactions (Continued)

Column Name	Datatype	Null
trans_type	NUMBER	Yes

Table 212: user_trans

Column Name	Datatype	Null
acct_id	NUMBER	Yes
data_id	NUMBER	Yes
data_tbl	NUMBER	Yes
end_dttm	DATE	Yes
start_dttm	DATE	Yes
type	NUMBER	Yes

Table 213: variables

Column Name	Datatype	Null
folder_id	NUMBER	Yes
indxtbl	NUMBER	No
interface_id	NUMBER	No
name	VARCHAR2	No
seq	NUMBER	Yes
val	VARCHAR2	Yes
var_id	NUMBER	No

Table 214: view_columns

Column Name	Datatype	Null
attributes	NUMBER	No
col_id	NUMBER	No
curr_id	NUMBER	Yes
data_type	NUMBER	No
display_order	NUMBER	Yes
group_order	NUMBER	Yes
heading	VARCHAR2	Yes
max_len	NUMBER	Yes
node_id	NUMBER	No
parsed	VARCHAR2	Yes
sort_direction	NUMBER	Yes
sort_order	NUMBER	Yes
val	VARCHAR2	No
val_col_refs	VARCHAR2	Yes
view_uid	NUMBER	No
width	VARCHAR2	Yes

Table 215: view_filter_list_items

Column Name	Datatype	Null
fltr_id	NUMBER	No
id	NUMBER	No

Table 215: view_filter_list_items (Continued)

Column Name	Datatype	Null
view_uid	NUMBER	No

Table 216: view_filters

Column Name	Datatype	Null
attributes	NUMBER	No
curr_id	NUMBER	Yes
data_type	NUMBER	No
fltr_id	NUMBER	No
fltr_name	VARCHAR2	No
fltr_type	NUMBER	No
oper_id	NUMBER	No
prompt	VARCHAR2	Yes
seq	NUMBER	No
val1	VARCHAR2	No
val1_col_refs	VARCHAR2	Yes
val1_parsed	VARCHAR2	Yes
val2	VARCHAR2	Yes
val2_col_refs	VARCHAR2	Yes
val2_parsed	VARCHAR2	Yes
view_uid	NUMBER	No

Table 217: view_join_filters

Column Name	Datatype	Null
attributes	NUMBER	No
curr_id	NUMBER	Yes
data_type	NUMBER	No
fltr_id	NUMBER	No
fltr_name	VARCHAR2	No
fltr_type	NUMBER	No
oper_id	NUMBER	No
tbl_id	NUMBER	No
val1	VARCHAR2	No
val1_col_refs	VARCHAR2	Yes
val1_parsed	VARCHAR2	Yes
val2	VARCHAR2	Yes
val2_col_refs	VARCHAR2	Yes
val2_parsed	VARCHAR2	Yes
view_uid	NUMBER	No

Table 218: view_node_filters

Column Name	Datatype	Null
attributes	NUMBER	No
curr_id	NUMBER	Yes
data_type	NUMBER	No

Table 218: view_node_filters (Continued)

Column Name	Datatype	Null
fltr_id	NUMBER	No
fltr_name	VARCHAR2	No
fltr_type	NUMBER	Yes
node_id	NUMBER	No
oper_id	NUMBER	No
val1	VARCHAR2	No
val1_col_refs	VARCHAR2	Yes
val1_parsed	VARCHAR2	Yes
val2	VARCHAR2	Yes
val2_col_refs	VARCHAR2	Yes
val2_parsed	VARCHAR2	Yes
view_uid	NUMBER	No
view_use	NUMBER	Yes

Table 219: view_nodes

Column Name	Datatype	Null
fltr_col_name	VARCHAR2	Yes
fltr_expr	VARCHAR2	Yes
fltr_expr_parsed	VARCHAR2	Yes
fltr_tbl_id	NUMBER	Yes
fltr_type	NUMBER	Yes
group_by_col_id	NUMBER	Yes
group_by_flag	NUMBER	No

Table 219: view_nodes (Continued)

Column Name	Datatype	Null
having_expr	VARCHAR2	Yes
having_expr_parsed	VARCHAR2	Yes
link_col_id	NUMBER	Yes
node_id	NUMBER	No
node_name	VARCHAR2	No
parent_node_id	NUMBER	Yes
starting_level	NUMBER	Yes
view_uid	NUMBER	No

Table 220: view_tables

Column Name	Datatype	Null
join_col_name	VARCHAR2	Yes
join_to_col_name	VARCHAR2	Yes
join_to_tbl_id	NUMBER	Yes
join_type	NUMBER	Yes
tbl	NUMBER	No
tbl_alias	VARCHAR2	No
tbl_id	NUMBER	No
view_uid	NUMBER	No

Table 221: views

Column Name	Datatype	Null
auto_refresh	NUMBER	Yes
created	DATE	Yes
fltr_expr	VARCHAR2	Yes
fltr_expr_parsed	VARCHAR2	Yes
folder_id	NUMBER	Yes
initial_search	NUMBER	Yes
interface_id	NUMBER	No
internal	NUMBER	Yes
last_mod	DATE	Yes
lines_per_page	NUMBER	Yes
name	VARCHAR2	No
owner_id	NUMBER	No
public_flag	NUMBER	Yes
seq	NUMBER	Yes
view_id	NUMBER	No
view_type	NUMBER	No
view_uid	NUMBER	No

Table 222: visibility

Column Name	Datatype	Null
admin	NUMBER	Yes

Table 222: visibility (Continued)

Column Name	Datatype	Null
cf_flags	NUMBER	Yes
enduser	NUMBER	Yes
id	NUMBER	No
interface_id	NUMBER	No
tbl	NUMBER	No

Index