

# Retek® Distribution Management 10.1



## Operations Guide – Volume 1: Functional Overviews



The software described in this documentation is furnished under a license agreement and may be used only in accordance with the terms of the agreement.

No part of this documentation may be reproduced or transmitted in any form or by any means without the express written permission of Retek Inc., Retek on the Mall, 950 Nicollet Mall, Minneapolis, MN 55403.

Information in this documentation is subject to change without notice.

Retek provides product documentation in a read-only-format to ensure content integrity. Retek Customer Support cannot support documentation that has been changed without Retek authorization.

**Corporate Headquarters:**

Retek Inc.  
Retek on the Mall  
950 Nicollet Mall  
Minneapolis, MN 55403  
888.61.RETEK (toll free US)  
+1 612 587 5000

Retek<sup>®</sup> Distribution Management<sup>™</sup> is a trademark of Retek Inc.

Retek and the Retek logo are registered trademarks of Retek Inc.

©2002 Retek Inc. All rights reserved.

All other product names mentioned are trademarks or registered trademarks of their respective owners and should be treated as such.

Printed in the United States of America.

**European Headquarters:**

Retek  
110 Wigmore Street  
London  
W1U 3RW  
United Kingdom  
Switchboard:  
+44 (0)20 7563 4600  
Sales Enquiries:  
+44 (0)20 7563 46 46  
Fax: +44 (0)20 7563 46 10



## ***Customer Support***

### **Customer Support hours:**

Customer Support is available 7x24x365 via e-mail, phone and Web access.

Depending on the Support option chosen by a particular client (Standard, Plus, or Premium), the times that certain services are delivered may be restricted. Severity 1 (Critical) issues are addressed on a 7x24 basis and receive continuous attention until resolved, for all clients on active maintenance.

<b>Contact Method</b>	<b>Contact Information</b>
-----------------------	----------------------------

<b>Internet (ROCS)</b>	<a href="http://www.retek.com/support">www.retek.com/support</a> Retek's secure client Web site to update and view issues
------------------------	--

<b>E-mail</b>	support@rettek.com
---------------	--------------------

<b>Phone</b>	US & Canada: 1-800-61-RETEK (1-800-617-3835) World: +1 612-587-5800 EMEA: 011 44 1223 703 444 Asia Pacific: 61 425 792 927
--------------	---

<b>Mail</b>	Retek Customer Support Retek on the Mall 950 Nicollet Mall Minneapolis, MN 55403
-------------	---

### **When contacting Customer Support, please provide:**

- Product version and program/module name.
- Functional and technical description of the problem (include business impact).
- Detailed step by step instructions to recreate.
- Exact error message received.
- Screen shots of each step you take.



# Contents

## Chapter 1 - System administration ..... 1

Introduction .....	1
Creating the rdmusr account on the operating system.....	1
Creating operating system accounts for all other users .....	4
Creating users in Oracle .....	4
Creating user accounts.....	4
Radio frequency operations .....	4
Operating system functions .....	5
Printer queues .....	5
Operating system scheduled jobs .....	6
Daemons .....	7
File management (directories) .....	7
System parameters.....	8

## Chapter 2 - DBA administration module ..... 23

Overview .....	23
Procedures .....	23
Open the Monitoring and Administration menu.....	23
Display locks on tables .....	24
Display table information .....	25
Display tablespace information .....	26
Display rollback information.....	27
Display index information .....	29
Display sequences information.....	30
Display the error log .....	31
View error log details .....	32
Delete error log records .....	33
Print the error log report .....	34

## Chapter 3 - RDM RIB components..... 35

Overview .....	35
Acronyms .....	35
Terms .....	36
Components.....	37
Vendor subscription.....	37
Location subscription .....	39
Item subscription .....	41
UDA subscription .....	46
Differentiator subscription.....	48
Purchase Order subscription.....	52

Inbound Work Order subscription .....	54
Inbound ASN subscription .....	56
Stock Order subscription .....	59
Outbound Work Order subscription .....	61
Pending Returns subscription .....	64
Inbound ASN publish .....	66
Appointments/Receipts publish .....	70
Stock Order Status publish .....	77
Outbound ASN publish .....	79
Inventory Adjustments publish .....	82
Inventory balance upload .....	85
Customer Returns publish .....	86
Return To Vendor publish .....	88
Streamsoft components .....	90
Space Locations .....	90
SKU OPTIMIZATION subscription .....	92
Items .....	94
<b>Chapter 4 - Subsystem interfaces .....</b>	<b>97</b>
Batch file formats .....	97
Unit pick system files .....	98
Allocation data download .....	98
Inbound carton download .....	98
Process UPS upload .....	99
Pick By Light interface .....	100
Files and directories .....	100
Download transactions .....	101
Upload transactions .....	105
Sortation subsystem interface .....	109
Files and directories .....	109
Download transactions .....	110
Upload transactions .....	111
Manifest mailing system .....	111
Files and directories .....	112
Views .....	112
Tables .....	114
Kewill shipping system Interface .....	116
Triggers .....	116
Packages .....	117
Tables .....	117
Rapistan Socket Interface .....	118
Triggers .....	118
Packages .....	118

Tables ..... 119





# Chapter 1 - System administration

## Introduction

Each system user must have a unique logon ID to the operating system, Oracle, and Retek Distribution Management. This section describes how to create each of these accounts.

## Creating the rdmusr account on the operating system

When Retek Distribution Management is first installed, the group rdm is created and the users rdmhost and rdmusr are created. The rdmusr user's home directory is typically /home/rdmusr and the rdmhost user's home directory is typically /home/rdmhost.

Add to the rdmusr profile script (.profile.) the following and set the protection to 740 (rwxr----):

```
ORACLE_SID=rdm;  
export ORACLE_SID  
TWO_TASK=rdm;  
export TWO_TASK  
TERM=vt220;  
export TERM  
ORACLE_TERM=vt220;  
export ORACLE_TERM  
ORACLE_HOME=/u01/app/oracle/product/dev6i;  
export ORACLE_HOME  
LD_LIBRARY_PATH=$ORACLE_HOME/lib;  
export LD_LIBRARY_PATH  
TNS_ADMIN=/u01/app/oracle/product/9i/network/admin;  
export TNS_ADMIN  
PATH=$ORACLE_HOME/bin:./:../:/usr/bin:/etc:/usr/sbin:/usr/ucb:/sbin:/usr/local/bin;  
export PATH  
TMPDIR=/tmp;  
export TMPDIR  
FORMS60_PLSQLV1_NAME_RESOLUTION=YES;  
export FORMS60_PLSQLV1_NAME_RESOLUTION  
FORMS60_OUTPUT=/tmp;  
export FORMS60_OUTPUT
```

```
FORMS60_PATH=/u01/app/rdm/bin;
export FORMS60_PATH
FORMS60_TERMINAL=/u01/app/rdm/bin;
export FORMS60_TERMINAL
REPORTS60_PATH=/u01/app/rdm/bin;
export REPORTS60_PATH
REPORTS60_TERMINAL=$REPORTS60_PATH;
export REPORTS60_TERMINAL
REPORTS_OUTPUT=/u01/app/rdm/reports;
export REPORTS_OUTPUT
RDM_BIN=/u01/app/rdm/bin;
export RDM_BIN
NLS_DATE_FORMAT="mm/dd/rr";
export NLS_DATE_FORMAT
NLS_LANG=American_America.UTF8
export NLS_LANG
RDMUSR=rdm username
    export RDMUSR
RDMPWD=rdm password
    export RDMPWD
menu.sh
exit
```

Add to the rdmhost profile script (.profile.) the following and set the protection to 740 (rwxr-----):

```
ORACLE_SID=rdm;
export ORACLE_SID
ORACLE_HOME=/u01/app/oracle/product/9i;
export ORACLE_HOME
TERM=vt220;
export TERM
ORACLE_TERM=vt220;
export ORACLE_TERM
TNS_ADMIN=/u01/app/oracle/product/9i/network/admin;
export TNS_ADMIN
LD_LIBRARY_PATH=$ORACLE_HOME/lib;
    export LD_LIBRARY_PATH
PATH=$ORACLE_HOME/bin:./:../:/usr/bin:/etc:/usr/sbin:
/usr/ucb:/sbin:/usr/local/bin;
```

```
export PATH
RDMUSR=rdm username
export RDMUSR
RDMPWD=rdm password
export RDMPWD
DOWNLOAD_DIR=/u01/app/rdm/hostcomm/download;
export DOWNLOAD_DIR
UPLOAD_DIR=/u01/app/rdm/hostcomm/upload;
export UPLOAD_DIR
SORTATION_DIR=/u01/app/rdm/hostcomm/sortation;
export SORTATION_DIR
```

**Note:** The value for fields shown in bold italics (above) must be set to the value appropriate for your installation.

The value of TERM is limited to the following choices:

- 1 ibm3151
- 2 vt220

### Creating operating system accounts for all other users

Create the user account in the operating system. The default shell should be ksh. The home directory should be /home/rdmusr, or whatever directory was assigned to the rdmusr. This prevents each user from having an individual home directory and makes the maintenance of the .profile easier.

Use whatever security measures are appropriate for your installation. You may use the operating system's security provisions for password expiration.

Retek Distribution Management enforces access control over the System screens regardless of the operating system security measures.

### Creating users in Oracle

Use Oracle's facility, such as Server Manager (svrmgrl), to create accounts in Oracle.

Set the default tablespace to USERS and the temporary tablespace to TEMP. The user's name must be the same as the account name on the operating system. The user's password must be the same as the user's name.

Grant the new user the wms\_user role. It has all the sufficient privileges to operate on all application tables.

Use the following SQL syntax to create new user accounts in Oracle:

```
create user rdmusr identified by <password>
default tablespace USERS
temporary tablespace TEMP;
grant wms_user to rdmusr;
```

### Creating user accounts

Add the user in Retek Distribution Management, using the User Table Editor screen. The password you specify in the system can be the same as or different than the user's operating system password.

### Radio frequency operations

When RF users are ready to begin work, they log into the operating system using their own account name and password. The .profile is executed, which sets the environment and executes the Retek Distribution Management application. The application takes the user's operating system account name as the Oracle account and password, and then starts the login to the system as that user. The first screen displayed is the Retek Distribution Management copyright. The login screen follows it. The user must fill in the username, password and facility ID.

## Operating system functions

This section explains how to set up print queues, describes the cron jobs and daemons, and gives instructions for file management.

### Printer queues

Create print queues in the operating system for reports and labels. Retek Distribution Management does not embed any printer-specific commands in jobs being sent to report printers. Label printer queues are typically defined as standard ASCII devices.

The names of the printer queues are specified on the System Parameter screen. Please also refer to the System Parameters section for more discussion of those parameters. The names of the parameters are listed here:

- 1 pick\_audit\_queue
- 2 pick\_label\_queue
- 3 pick\_package\_queue
- 4 recv\_audit\_queue
- 5 recv\_label\_queue
- 6 recv\_receipt\_queue
- 7 reprint\_label\_queue
- 8 ship\_bol\_queue
- 9 ship\_label\_queue
- 10 unit\_pick\_lbl\_queue

## Operating system scheduled jobs

This table describes the programs that should be run periodically to remove obsolete data from the system, to schedule locations for cycle counting, and to close appointments that are now reconciled. You should run these programs using the operating system facility (cron) for scheduling jobs for unattended operation.

In the table, the name of the program to run is listed under the column heading "Name." The programs are in the \$RDM\_BIN directory. For each routine, a system parameter exists that specifies the number of days of data to retain. These parameters are maintained on the System Parameter editor, which is described in the Retek Distribution Management User Guide.

rdmhost user must submit the jobs. The user's cron should first execute the .profile to set the environmental variables.

Name	Purpose	Parameters	Frequency
dc_view.sh	Refreshes data in the DC VIEW and DC UTILIZATION tables	facility_id	Once daily
inv_bal_upload_b.sh	Create an inventory balance upload file for each facility id, part of the facility type	facility_id	Once daily
maintain_wave_stats.sh	Updates wave statistics		Every 15 minutes
purge_appointments_b.sh	Cleans the Appointment and related tables.	Appointment_window & appt_purge_days	Once daily
purge_labor_prod_b.sh	Cleans the Labor Productivity table.	Purge_lab_prod_days	Once daily
purge_manifest_b.sh	Cleans the Manifest and related tables.	manifest_purge_days	Once daily
purge_rtv_b.sh	Cleans the Return to Vendor table.	Purge_rtv_days	Once daily
purge_uploads_b.sh	Cleans the upload tables.	Upload_purge_days	Once daily
purge_vendor_trouble_.sh	Cleans the vendor trouble history tables.	Trouble_purge_days	Once daily
run_distribution.sh	Matches inventory to allocation requests, creates pick directives and prints picking packages (if applicable)		Manually or every 15 minutes
schedule_cycle_count_b.sh	Schedules cycle counts (SS) for the DC.	Cycle_count_period	Once daily

Name	Purpose	Parameters	Frequency
schedule_rop_distribution.sh	Schedules a Re-Order Point distribution run.		Every 5 minutes
unreconciled_appt_monitor.sh	Closes any unreconciled appointments that have had all labels scanned or nulled.	None	Hourly

## Daemons

One daemon process must be run continuously. It should be run by the user rdmhost from the \$RDM\_BIN directory. The Calling Syntax includes the parameter -s, which is the sleep time in seconds: how often the daemon should wake up and look for inducted or diverted cartons. A typical value is between 10 and 30 seconds. In the Calling Syntax, <user\_name/password> refers to an Oracle user and password.

Name	Description	Calling Syntax
<b>Read_divert_data</b>	Loads the sorter intake table from a data file.	read_divert_data <username/password> <facility_id>-s[n]

## File management (directories)

Discusses permissions and any file cleanup (purging needed for each directory).

Directory	Path	Purpose	Perm	Purging
<b>Base Directory</b>	\$RDM	This is the base directory that other directories will branch from	775	None
<b>Reports</b>	\$RDM/reports	Temporary holding area for reports (line and label). Any report sent to file remains here.	777	Occasional (weekly)
<b>Host Download</b>	\$DOWNLOAD_DIR	Temporary holding area for files to download and log files.	775	Occasional (weekly)
<b>Host Upload</b>	\$UPLOAD_DIR	Temporary holding area for files to upload.	775	Occasional (weekly)
<b>Sortation</b>	\$SORTATION_DIR		775	Occasional (weekly)
<b>BIN</b>	\$RDM/bin	Holds all executables.	755	None
<b>INSTALL</b>	\$RDM_ADMIN/create	Holds files used to build the system.	755	None

## System parameters

Each facility in the DC has its own set of system level parameters. You can view and modify these in the System Parameter screen.

This table gives the name of each parameter, briefly explains its purpose and how it is used, and gives the allowable entry type, as described in this list:

**Activity:** Activity code, found in service\_standards table.

**Dest ID:** Destination ID, found in Ship Destination table.

**Fixed:** Cannot be modified by the user.

**Item ID:** Item ID, found in Item Master table.

**Location:** Location ID, found in Location table.

**Location Type:** Location Type found in the Loc\_type table

**Number:** Numeric value.

**Queue:** Printer (line or label) queue.

**Text:** Free form text.

**Time:** Valid time (24 hour format - HH:MI).

**Wip:** Wip code, found in the wip\_codes table.

**Y/N:** “Y” = Yes, “N” = No.

Name	Purpose	Type
DC_dest_ID	Destination ID of the DC. Must be in the Ship Dest table. Used in reports (for DC return address) and to show what containers are stock (dest_ID=DC).	Dest ID
LTC	Unit Pick System Code associated to the RF Unit Picking (Stationary SKU)(LTC and ltc code refer to the same operation of Less Than Case picking)	Text
PPS	Unit Pick System Code associated to a Paperless Picking System(PPS and pps code refer to the same paperless picking system)	Text
PPS_flag	Indicates whether PPS is turned on. Used in the distribution and picking processes.	Y/N
TASK_OPT	Specifies the ordering of assigned tasks.	TEXT or Location_id
able_to_ship_level	Security level to enable the F9 ship key in the shipping form.	Number



Name	Purpose	Type
active_ovrszd_putwy	Default putaway plan for an oversized item (when no putaway plan specified in Item Master)	Putaway Plan
adjust_pick	Enable the F7 adjust key on the RF picking screens.	Y/N
ahl_log	Log Activity History Log. 0: No AHL Logging 1: AHL Logging through SQL insert 2: AHL Logging through Oracle Queues	0, 1, 2
allow_bulk_pick	When set to 'N', the distribution process bypasses the assignment of Bulk Picks and proceeds to the Container Picking logic.	Y/N
allow_bulk_replen	When set to 'N', the distribution process bypasses the assignment of Bulk Replenishment Picks and goes to the Container Replenishment Picking logic.	Y/N
allow_cont_replen	When set to 'N', the distribution process bypasses the assignment of Container Replenishment Picks.	Y/N
allow_container_pick	When set to 'N', the distribution process bypasses the assignment of Container Picks and goes to the Unit Picking logic.	Y/N
allow_rtn_replace	When set to 'Y', RDM allows item replacement and displays a screen to the user where an alternate item is entered to replace the returned one.	Y/N
allow_trble_putaway	Allows the Putaway screen to complete the putaway of a container that has a Trouble Code associated with it.	Y/N
apply_qa_wip	Determines if a QA Wip needs to be applied	Y/N
appointment_window	The number of days (past and future) to allow appointments to be active. Used in the Schedule Appointment screen and purge_appointments_b.sh.	Number
appt_purge_days	Number of days after closure to purge an appointment. Used in purge_appointments_b.sh.	Number

Name	Purpose	Type
assortment_wip_code	Wip Code applied when inbound container has an assortment item. Parent Item with child SKUs.	Wip
auto_induct	When set to 'Y', groups assigned to the first pack wave have the Active Pick flag set to 'Y', indicating that this pack wave is to be staged in the UPS for picking. If put_to_order is enabled, allocation data is sent to the UPS for only those allocations deemed active within the UPS. If put to destination is enabled, all allocations are downloaded at one time.	Y/N
autopack	Assigned name to the Autopack Sorter	Text
back_order_flag	Indicates whether to retain stock orders when the inventory is exhausted. Used in the distribution process.	Y/N
best_before_wip	Used to automatically apply a wip code to a container requiring a best before date (perishable indicator set in Item Master)	Wip
break_by_wip_con	When set to 'Y', the distribution process creates separate Master Pick Labels for each group of WIP Codes for conveyable cartons.	Y/N
break_by_wip_non_con	When set to 'Y', the distribution process creates separate Master Pick Labels for each group of WIP Codes for non-conveyable cartons.	Y/N
carton_store_putwy	Default putaway plan for a single container	Text
company_nbr	Company number to send to PPS.	Number
consolidate_pend_wip	When set to 'Y', RDM allows the consolidation of WIP codes, when building pallets.	Y/N
container_format	Indicates that the container identifier number is compliant with UCC128 or is generic with embedded destination ID.	UCC128 or default
cs_rsv_loc_type	User Defined location type for case reserve	Loc Type
cs_rsv_priority	Priority used in distribution to pull merchandise from case	Number
cycle_count_period	Number of days to cycle count the entire DC. Used in schedule_cycle_count_b.sh.	Number

Name	Purpose	Type
cycle_count_type	Defines how the DC wants to count inventory, either by item, location or zone. Used when schedule cycle count runs in cron (System Scheduled Cycle Count)	Text
def_bulk_replen_res	Sets the number of Bulk Replenishment resources to use to display the Wave Duration on the wave planning screens.	Number
def_bulk_resources	Sets the number of Bulk resources to use to display the Wave Duration on the wave planning screens.	Number
def_cont_resources	Sets the number of Container Pick resources to use to display the Wave Duration on the wave planning screens.	Number
def_cont_replen_res	Sets the number of Container Replenishment Pick resources to use to display the Wave Duration on the wave planning screens.	Number
def_prime_urpln_res	Sets the number of prime Replenishment Pick resources to use to display the Wave Duration on the wave planning screens.	Number
def_unit_replen_res	Sets the number of Unit Pick Replenishment resources to use to display the Wave Duration on the wave planning screens.	Number
def_unit_resources	Sets the number of Unit Pick resources to use to display the Wave Duration on the wave planning screens.	Number
def_work_day_end	Default working day end. Used in Working Days Editor.	Time
def_work_day_start	Default working day start. Used in Working Days Editor.	Time
default_carton_group	Carton group used in cartonization if none is defined for the item.	Text
default_cc_plan	default cycle count plan to be set during item master download if none is specified.	Text
default_kitting wip	WIP Code when creating an item that is defined as a kit	Wip

Name	Purpose	Type
default_order_level	Type of saved query for order selection.	FULL – Every line in query has to match. ORDER – If any line matches, RDM will show all distro lines. LINE – Only distros that match display.
default_order_type	Determines how stock orders that are downloaded are processed. WAVE uses the pre-defined Shipping Schedule and proceeds without intervention. AUTOMATIC does not require destinations to have a pre-defined Shipping Schedule, but does proceed without intervention. MANUAL allows intervention by selecting orders to be included in a wave. PO – allocation of merchandise is tied to a specific PO. PREDIST – allocations that have predistributed merchandise.	WAVE, AUTOMATIC, MANUAL, PO and PREDIST
default_putaway	default putaway plan to be set during item master download if none is specified	Text
default_trailer_cube	Default size of a trailer. Used in the Schedule Appointment screen when a new trailer is scheduled. Used to calculate and display the percentage filled of a trailer on the Shipping Status.	Number
default_ups	Default Unit Pick System code for Item Master download	Text
delete_pfl	When set to 'Y', and unit quantity in the 'from location' is 0, RDM will delete the location record once the merchandise is moved out. User override is provided on the FPL Move screen.	Y/N
display_item_id	Used in the multi-item UPC functionality. If set to 'Y', item information, which matches the UPC code, will display when the item_id is scanned. When set to 'N', only the UPC code displays in the field.	Y/N

Name	Purpose	Type
distrib_unfin_wip	When set to 'Y', RDM allows allocation of merchandise from a pallet that has unfinished WIP codes associated with it.	Y/N
distribute_partial	When set to 'Y', RDM will process partial distribution of a dye lot. The maximum amount of a single dye lot will be distributed even if only a partial fulfillment of the order. If 'N', the distribution is skipped.	Y/N
drop_off_convey	location to be displayed during picking drop off if conveyors are used in facility.	Location
dynamic_random_slot	determine whether distribution should create a random slot for active picking when needed.	Y/N
enable_kitting	When set to 'Y', the Distribution process builds Kit Build directives for Master Items that have a Stock Allocation but no Inventory to satisfy the order.	Y/N
enforce_prime_relabl	Re-labeling for Prime Picking	Y/N
exceed_capacity	allows chutes to be overfilled during the waving process	Y/N
exception_cont_type	default container type to be used during cartonization if no defined container types will hold items	Container Type
exceptions_stage	area specified in building (location) where exception packages should be sent for consolidation	Location
first_time_sku	wip code to be applied to first time SKU containers during receiving	Wip
fixed_replen_wave	When set to 'Y', RDM groups all replenishment picks into Wave 1. When set to 'N' RDM associates replenishment picks with the wave that originated the need.	Y/N
fpl_replen_dest_id	Destination identifier used for replenishing of Forward Pick Locations when replenishment method is PREPLANNED.	Dest ID
fstsku_bypass_fl	Indicates to conveyor system to weigh or not weigh a carton with first time SKU wip applied.	Y/N

Name	Purpose	Type
generate_rma	When set to 'Y', the distribution process generates a unique number that is assigned per container. This generation process happens after the 'pick-to' containers are split out based on volumetric data. When set to 'N', RMA numbers are not generated.	Y/N
gift_card_wip	Defined WIP code denoting containers that require the insertion of a specialized gift card.	Wip
gift_w_wip	Defined WIP code for gift wrapping	Wip
group_picks_active	Determine how distribution should cartonize active picks	Y/N
hold_first_time_sku	wip applied to all like containers for items where one container has first time SKU wip applied	Wip
hot_replen_putaway	When set to 'Y', Putaway looks for Unit Replenishment opportunities.	Y/N
hot_replen_recvg	When set to Y, receiving allocation process looks for Unit Replenishment opportunities. When set to N, receiving allocation process functions as normal.	Y/N
in_transit_loc	Location of containers in process. Used in Move, Putaway, and Picking screens.	Location
interleaved_cc	When set to 'Y', RDM suggests a location for system scheduled cycle count after a putaway operation. When set to 'N', Putaway and Cycle Count task are not interleaved.	Y/N
kitting_activity_code	Activity code associated with kitting against which statistics are collected.	Activity
ltc_code	Unit Pick System Code associated to the RF Unit Picking (Stationary SKU)(LTC and ltc code refer to the same operation of Less Than Case picking)	Text
labeled_picking	When set to 'Y', RDM generates a picking label packet and a report. When set to 'N', RDM assumes labelless picking and only generates a report.	Y/N

Name	Purpose	Type
labeled_receiving	When set to 'Y', RDM generates a receiving label packet and a report. When set to 'N', RDM assumes labelless receiving and only generates a report.	Y/N
labeled_reserve	When set to 'Y', RDM tracks each container in reserve storage with a separate identifying label. When set to 'N', only master containers in reserve are labeled.	Y/N
load_sequencing	When set to 'Y', RDM sorts picks with respect to the defined route/destination load sequence. When set to 'N', RDM sorts according to distro number sequence.	Y/N
log_interface_error	Determines whether RDM Interface APIs log an error using the log_oracle_error function when an error occurs. Note: this must be set to N in an enterprise/SeeBeyond environment because of Oracle distributed processing and support for AUTONOMOUS TRANSACTIONS	Y/N
manifest_purge_days	Number of days after shipping to purge a manifest. Used in purge_manifest_b.sh.	Number
max_group_units	used with group picks active. Numeric values that sets max number of units to be allocated to one group	Number
max_wave_nbr	Maximum wave number allowed to be maintained in the distribution screens.	Number
max_wave_rows	Maximum number of orders/rows that may be retrieved from a specific query. This number is used when the user does not include the max number as part of a query.	Number
min_auto_wave	The lowest wave number that can be used by RDM when assigning orders. The system assigns any orders retrieved by a specific query to the first wave with the status of AVAIL, type of MANUAL and greater than or equal to the min_auto_number.	Number
mixed_dest_id	Destination ID where containers holding merchandise for different destinations are sent for separation..	Dest ID

Name	Purpose	Type
mixed_wip_stage_loc	Location identifier at which containers with different WIP codes are staged for separation.	Location
mm_allow_distrib	determines whether or not distribution is allowed to distribute from manually marked locations	Y/N
multi_open_manifest	When set to 'Y', indicates that multiple destinations can be actively loaded into a single trailer simultaneously.	Y/N
multi_sku_wip	wip code applied to inbound container that contain more than one container item record	Wip
nbr_cartons_pallet	Max number of cartons per pallet, in putaway logic.	Number
nbr_items_pallet	Max number of items per pallet, in putaway logic.	Number
nbr_skus_per_pallet	Max number of SKUs per pallet, in putaway logic.	Number
order_line_number	Y if orders are being tracked at the order line level	Y/N
order_set_stage	location in facility where outbound cartons will be directed to have order sets printed.	Location
order_status_upload	Y if order status information will be uploaded to the host	Y/N
pack_lane_stage	staging location where outbound orders are sent to be packed	Location
pack_wave_stage	staging location where cartons are sent to await induction into a unit sorter	Location
pallet_flow_loc_type	user defined location type for pallet flow reserve	Location Type
pallet_flow_priority	priority used during distribution to pull merchandise from case reserve	Number
pallet_rsv_loc_type	user defined location type for pallet reserve	Location Type
pallet_rsv_priority	priority used during distribution to pull merchandise from case reserve	Number
pallet_store_putwy	default putaway plan to be used for items that do not have a putaway plan specified	Text



Name	Purpose	Type
password_expire	Number of days since the last password change; forces users to change their password.	Number
password_old	Number of days since the last password change; suggests that users change their password.	Number
pbl_pick_to_reserve	When set to 'Y', causes the system to generate a distribution detail record to download to the Pick-By-Light system, which will cause the excess units to be re-boxed and returned to inventory. This parameter is applicable only when the pps_round_up flag is set to 'N'.	Y/N
pbl_replen_dest_id	Default destination assigned for replenishment to the PPS system.	Dest ID
pend_first_time_sku	Cartons of an item on a receipt to be held on the receiving dock until the first time SKU wip is removed	Y/N
pick_audit_queue	Line printer queue where the Pick Audit List prints.	Queue
pick_by_loc_flag_con	When set to 'Y', RDM is picking by location and permits mixing of conveyable cartons of varying destinations onto a single pallet during Container Picking. When set to 'N', RDM is picking by destination and does not permit mixing of conveyable cartons of varying destinations onto a single pallet during Container Picking.	Y/N
pick_by_loc_flag_non	When set to 'Y', RDM is picking by location and permits mixing of non-conveyable cartons of varying destinations onto a single pallet during Container Picking. When set to 'N', RDM is picking by destination and does not permit mixing of non-conveyable cartons of varying destinations onto a single pallet during Container Picking.	Y/N
pick_existing	Determines whether or not to include the inbound quantity associated to a forward pick location when determining amount of units available for picking	Y/N
pick_label_queue	Label printer queue where the pick labels will print.	Queue

Name	Purpose	Type
pick_label_set	Determines the scheme used to generate the Container Identifier.	UCC128 or default
pick_printer_type	Printer description	Text
pnad_isd_lead_time	Pick not after date/In store date lead time.	Number
pps_code	Unit Pick System Code associated to a Paperless Picking System(PPS and pps code refer to the same paperless picking system)	Text
pps_drop_off_loc	Location where containers bound for PPS are dropped off.	Location
pps_pickup_loc	Location at which the system picks up cartons packed by PPS	Location
pps_round_up	When set to 'Y', the distribution process will increase (round up) the distribution evenly across the destinations to consume the excess. When set to 'N', the process will not exceed the requested quantity. The parameter pbl_pick_to_reserve is applicable only when the pps_round_up flag is set to 'Y'.	Y/N
pre_manifest_bol	Default sequence number for pre manifest BOLs. Used in the Conveyor Cutoff and Ship Trailer screens.	Number
preplan_unit_replen	When set to 'Y', unit picks are planned to replenish the entire wave's needs during the Distribution Process. When set to 'N', RDM assumes the use of Re-order Point (or Max/Min) Replenishment.	Y/N
prime_pick	Y/N indicator determining if the location is eligible for prime picking.	Y/N
prime_pick_dropoff	Location for prime pick replen drop.	Location
print_and_apply	Location where print and apply labels occurs	Location
pts_ctn_max_days	Number of days before open Put To Store carton is flagged.	Number
purge_RTV_days	The number of days after a Return to Vendor to purge an RTV. Used in purge_rtv_b.sh.	Number

Name	Purpose	Type
purge_act_based_cost	The number of days after activity based cost figures are calculated to purge the ABC data. Used in purge_activity_based_cost.sh	Number
purge_lab_prod_days	The number of days after activities to purge labor productivity data. Used in purge_labor_prod_b.sh.	Number
putaway_stage_loc	Default location suggested for a two-step putaway.	Location
pts_loc_type	Default location type for Put To Store	Location Type
qa_bypass_fl	Indicates if sortation system should weigh an inbound carton that has a QA wip applied	Y/N
qa_to_active	Allow cartons with QA wips to be sent directly to active. Works in conjunction with hot_replen_recvg	Y/N
qa_wip_code	Wip code to be applied to cartons that need an inbound QA	Wip
qc_audit_queue	Printer queue where the Quality Audit prints.	Text
qlty_activity_code	Activity Code for the Quality Audit operation.	Activity
quality_wip_code	Defined WIP Code applied to cartons during the Prereceiving Process to mark for Quality Audit.	Wip
random_active_stage	Staging location where replenishment containers for random active are placed.	Location
random_repln_dest_id	Destination ID for Random Active locations	Dest Id
reassign_wip	Defined WIP code that reassigns a group of containers from a single destination to another single destination.	Wip
receipt_level	Determines the level at which the receipt uploads will be processed. Valid values are 'A'ppointment and 'C'ontainer	Text
recv_audit_queue	Line printer queue where the Receiving Audit List will print.	Queue
recv_label_queue	Printer queue where the Receiving Label Package is printed.	Queue

Name	Purpose	Type
recv_label_set	Format of the Container Identifier used when generating Receiving Labels.	UCC128 or default
recv_printer_type	Name of the printer and the size label stock that matches the label definition.	Text
recv_receipt_queue	Label printer queue where the receiving labels print.	Queue
reg_pack_chute	chute designator for regular packing chutes	Text
replen_from_prime	Indicates whether or not replenishment picks can be created from the prime pick area.	Y/N
replenishment_level	When a unit picking location is expected to drop below this value multiplied by its units capacity, the system generates a replenishment pick. Used in the distribution process and when preplanned_unit_replen scp parameter = 'Y'..	Number
reprint_label_queue	Printer queue where the labels generated by the Reprint/Null Labels screen are printed.	Queue
reprint_printer_type	Name of the printer and the size label stock that matches the label definition.	Text
reserve_ovrszd_putwy	Putaway plan for oversized cartons	Text
retain_label_file	Indicates whether the label print file that was sent to the printer is kept in the \$ RDM/reports directory.	Y/N
reticketing_wip_code	Defined WIP code denoting containers that need new retail price tags.	Wip
return_replace_code	Defined WIP codes denoting a returned container that holds items requiring replacement.	Wip
return_to_vendor_loc	Location ID that identifies the location where return to vendor processing takes place.	Location_id
return_wip	Defined WIP codes that denotes a returned container.	Wip
returns_location	Location ID that identifies the location where returns processing takes place.	Location_id

Name	Purpose	Type
rf_asn_position	Determines the starting position for display of the ASN Number on the RF screens	Number
rf_item_position	Determines the starting position for display of the item id on the RF screens	Number
rop_dist_method	Method of distribution to be used for re-order point replenishment. Choose from PICKTOCLEAN, FIFO and EFFICIENCY.	Text
ship_bol_queue	Line printer queue where the Bill of Lading prints.	Queue
ship_label_queue	Printer queue where shipping labels print.	Queue
ship_logical_pallet	Logical Pallet in Shipping	Text
ship_printer_type	Type of printer at which shipping labels are printed.	Text
singles_sorter_group	Sorter group defined for Singles processing	Text
ticketing_wip_code	WIP Code to apply for ticketing processing.	Wip
trans_wip_in_to_out	Determines whether any inbound work orders associated to a PO/Item should be applied to cross-docked containers and processed as outbound work orders	Y/N
trouble_purge_days	Number of days to retain on file for Container and Appointment History.	Number
ucc_container_app_id	Specific_business ID for use with UCC128 label generation.	Text
ucc_container_org_id	Value to use when creating an UCC128-compliant carton serial number.	Text
ucc_manufacturer_id	Value to use when creating an UCC128-compliant carton serial number.	Text
unit_block_dist_flag	When set to 'Y', RDM distributes units in Block. Block indicates that shortages are borne by the lower priority destinations. When set to 'N', RDM distributes units in Round Robin. Round Robin spreads shortages proportionally among all destinations. Used in the distribution process for LTC locations only.	Y/N

Name	Purpose	Type
unit_pick_lbl_queue	Printer queue where packing slip prints. Used in the Select Orders screen for unit picks only.	Queue
unit_picking_flag	When set to 'N', the distribution process bypasses the assignment of Unit Picks.	Y/N
unknown_item	Item ID of unknown merchandise. Used in the Build Container screen.	Item ID
unknown_rma	Generic ID for returned containers that do not include the original RMA number.	Item ID
unlocated_location	Location of lost containers, those that cannot be found during a cycle count. Used in the Count Location screen.	Location
upld_convert_inv_adj	When set to 'Y', RDM uploads an inventory adjustment when converting inventory to inventory during startup.	Y/N
upload_purge_days	The number of days after an upload to purge the upload data. Used in purge_uploads_b.sh.	Number
usps_priority_code	Default Service Code for the Pack Slip	Text
usps_service_code	Default Route for the Pack Slip	Text
va_wip_code	WIP code used for when Vendor Assurance	Wip
version_number	Number of the System version.	Fixed
virtual_distro	Distro number assigned to unreconciled store orders from a Unit Pick System	Text
weigh_wip_code	Defined WIP code that assigns a WIP code to weigh merchandise that has a catch weight.	Wip
work_on_saturday	When set to 'Y', RDM sets Saturday as a working day. Used in the Working Days Editor.	Y/N
work_on_sunday	When set to 'Y', RDM sets Sunday as a working day. Used in the Working Days Editor.	Y/N
xzone_pick	When set to 'Y', the distribution process creates pick across multiple zones for the same distro. When set to 'N', cross-zone picking, for the same distro, is denied	

## Chapter 2 - DBA administration module

### Overview

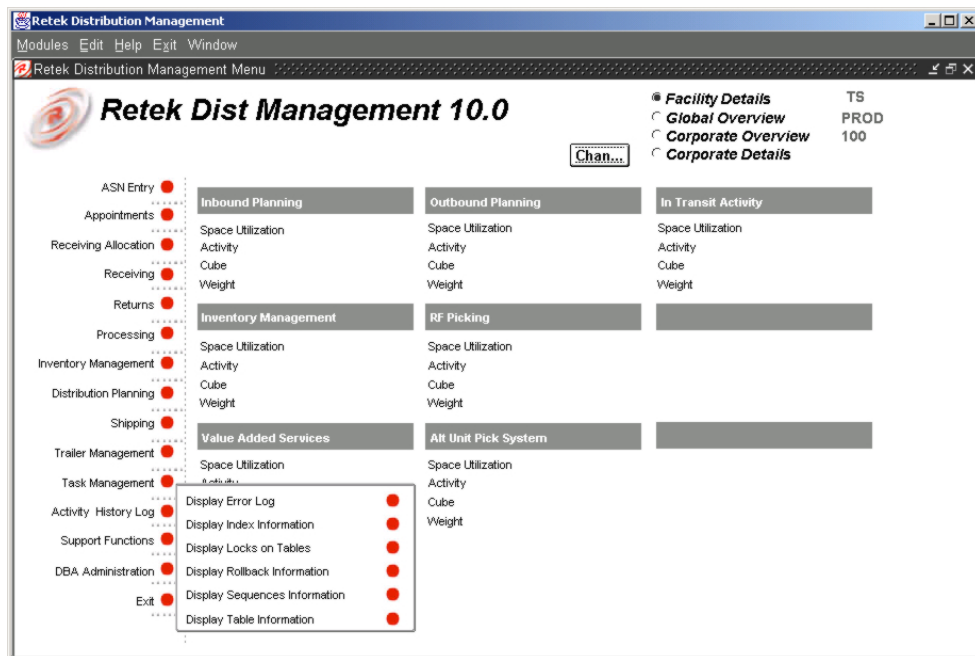
The DBA Administration module allows the DBA Administrator to monitor database information such as table locks, tablespace, indices, and errors.

The Administration section describes how to display locks on tables, table information, tablespace information, rollback information, index information, sequence information, and the error log.

### Procedures

#### Open the Monitoring and Administration menu

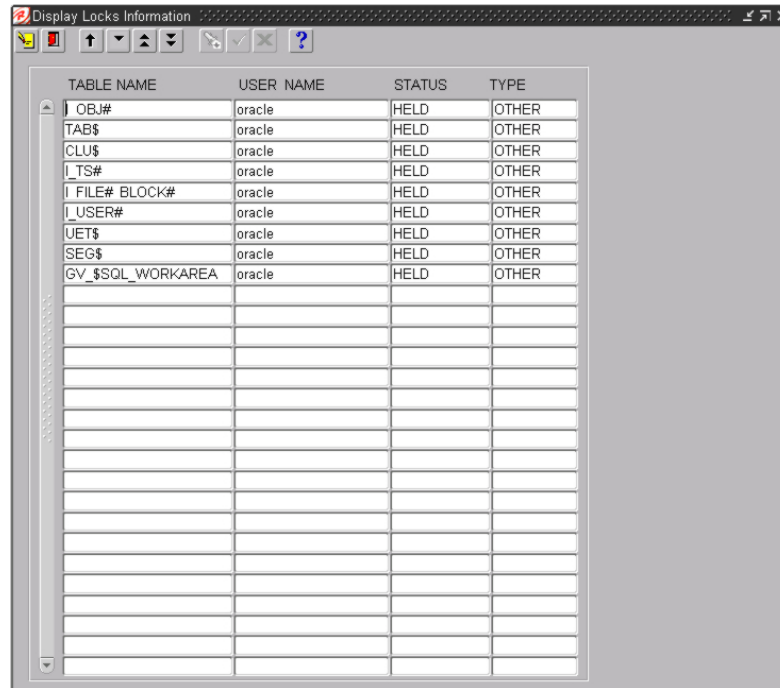
- 1 Select the DBA Administration menu.



*Monitoring and Administration menu*

## Display locks on tables

- 1 On the DBA Administration menu, select the Display Locks on Tables option.
- 2 The Table Locks Monitoring screen is displayed.



### Table Locks Monitoring Screen

- 3 Click the **Refresh** button to view the table locks up-to-the-minute status.

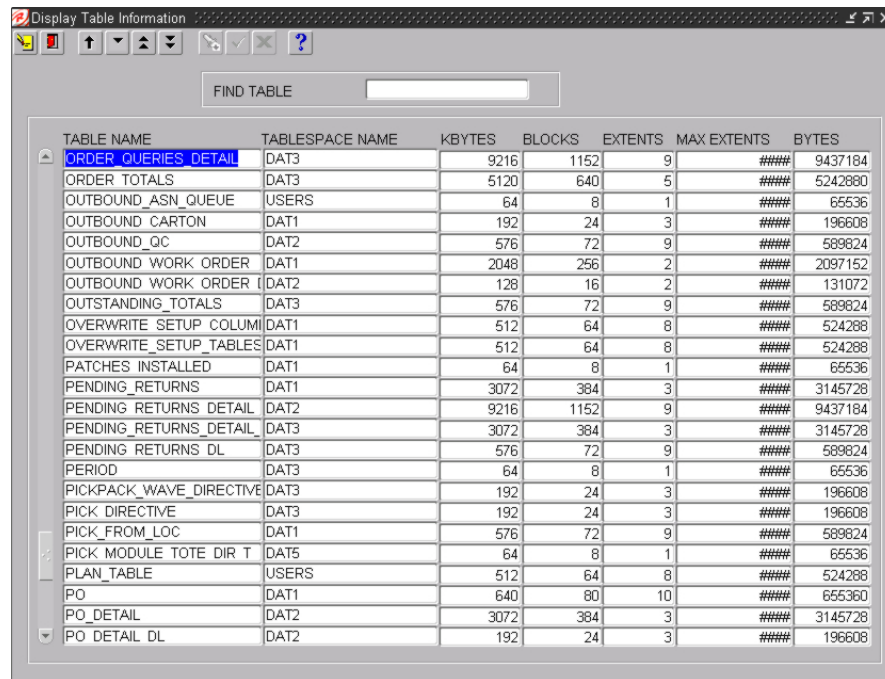


## Display table information

You can use the Display Table Information option to display specific table information.

**Note:** You can monitor the number of extents to detect table growth. A large extent value indicates possible table fragmentation. If the number of extents approaches the maximum, the table should be rebuilt.

- 1 On the DBA Administration menu, select the Display Table Information option.
- 2 Click the **Display** button. The Table Information screen is displayed.



The screenshot shows a window titled "Display Table Information" with a toolbar at the top containing icons for search, navigation, and help. Below the toolbar is a "FIND TABLE" search box. The main area displays a table with the following columns: TABLE NAME, TABLESPACE NAME, KBYTES, BLOCKS, EXTENTS, MAX EXTENTS, and BYTES. The table lists various database tables, with "ORDER\_QUERIES\_DETAIL" selected and highlighted in blue.

TABLE NAME	TABLESPACE NAME	KBYTES	BLOCKS	EXTENTS	MAX EXTENTS	BYTES
ORDER_QUERIES_DETAIL	DAT3	9216	1152	9	####	9437184
ORDER_TOTALS	DAT3	5120	640	5	####	5242880
OUTBOUND_ASN_QUEUE	USERS	64	8	1	####	65536
OUTBOUND_CARTON	DAT1	192	24	3	####	196608
OUTBOUND_QC	DAT2	576	72	9	####	589824
OUTBOUND_WORK_ORDER	DAT1	2048	256	2	####	2097152
OUTBOUND_WORK_ORDER_I	DAT2	128	16	2	####	131072
OUTSTANDING_TOTALS	DAT3	576	72	9	####	589824
OVERWRITE_SETUP_COLUMNS	DAT1	512	64	8	####	524288
OVERWRITE_SETUP_TABLES	DAT1	512	64	8	####	524288
PATCHES_INSTALLED	DAT1	64	8	1	####	65536
PENDING_RETURNS	DAT1	3072	384	3	####	3145728
PENDING_RETURNS_DETAIL	DAT2	9216	1152	9	####	9437184
PENDING_RETURNS_DETAIL_DL	DAT3	3072	384	3	####	3145728
PENDING_RETURNS_DL	DAT3	576	72	9	####	589824
PERIOD	DAT3	64	8	1	####	65536
PICKPACK_WAVE_DIRECTIVE	DAT3	192	24	3	####	196608
PICK_DIRECTIVE	DAT3	192	24	3	####	196608
PICK_FROM_LOC	DAT1	576	72	9	####	589824
PICK_MODULE_TOTE_DIR_T	DAT5	64	8	1	####	65536
PLAN_TABLE	USERS	512	64	8	####	524288
PO	DAT1	640	80	10	####	655360
PO_DETAIL	DAT2	3072	384	3	####	3145728
PO_DETAIL_DL	DAT2	192	24	3	####	196608

**Table Information Screen**

These are the fields on the Table Information screen:

Field name	Field description
Find Table	Table name for table to be queried.
Table Name	Name of the database table.
Tablespace Name	Tablespace name.
Kbytes	Number of (K) bytes in the table.
Blocks	Number of blocks the table is using.
Extents Cur.	Current table extents.
Extents Max	Maximum allowable table extents.

- 3 Click the **Display** button at the blank Find Table field to display a list of all tables.

**Note:** If you want to display information about a particular table, enter the specific table name at the Find Table field. You can also enter a partial table name. For example, you can enter ‘APP’ to display all tables that begin with these letters.

- 4 Click the **Exit** button to return to the DBA Administration menu.

## Display tablespace information

You can also use the Display Tablespace Information option to display tablespace specific information, such as the amount of free space in a tablespace or the number of extents in a table space.

- 1 On the Monitoring and Administration menu, select the Display Tablespace Information option.
- 2 Click the **Display** button. The Tablespace Information screen is displayed, along with all tablespace.

[illegible]

### Tablespace Information Screen

These are the fields on the Tablespace Information screen:

Field name	Field description
Tablespace Name	Name of the database tablespace.
File Name	Name of the data file.
Mbytes	Table size in mega bytes.
Status	Indicates whether tablespace is Available or offline.

- 3 Click the **Refresh** button to refresh the screen and view any new tablespace information.

## Display rollback information

You can use the Display Rollback Information option to display information about rollbacks. You can also use this information to determine whether the rollback segments need to be enlarged for a specific installation.

- 1 On the DBA Administration menu, select the Display Rollback Information option.
- 2 The Rollback Information screen is displayed, along with all rollback segments.

[illegible]

### *Rollback Information Screen*

These are the fields on the Rollback Information screen:

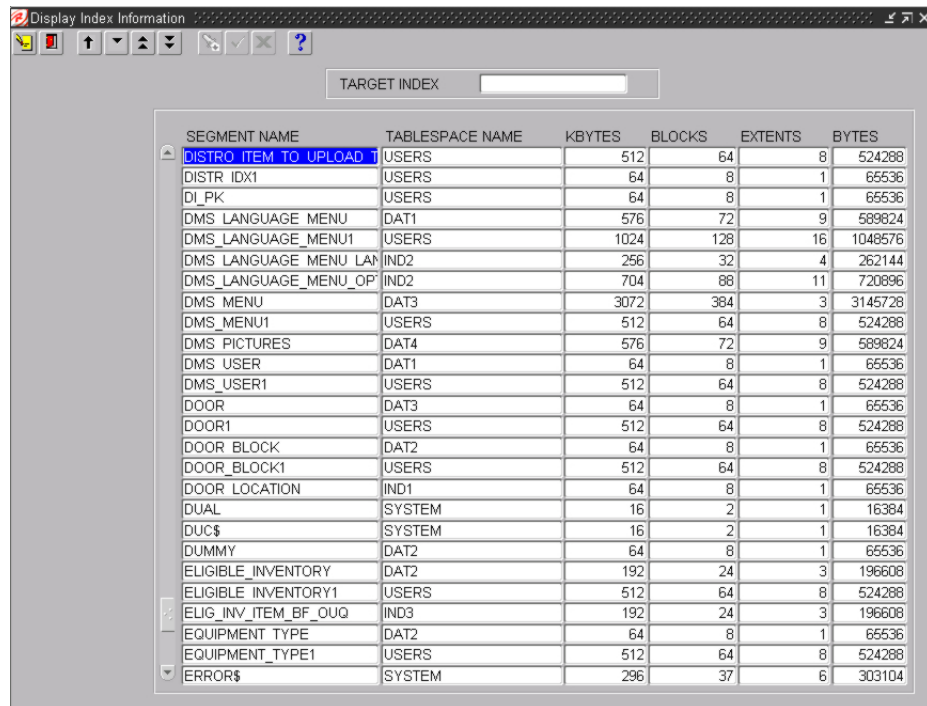
Field name	Field description
Rollback Segment	Name of the rollback segment.
Increase RB	Indicates when additional rollback segments need to be added. YES = rollback segments should be added. NO = rollback segments do not need to be added.
KSize	Size of rollback segments in bytes.
Extents	The number of times that the rollback segment had to acquire a new extent
XACTS	Number of Active Transactions
WAITS	The number of rollback segment header requests that resulted in waits
GETS	The number of rollback segment header requests
OPTSIZE	The value of the optimal parameter for the rollback segment
STATUS	Status (Online / Offline)
RRSIZE	Rollback Size

- 3 Click the **Refresh** button to refresh the screen and view any new information.
- 4 Click the **Exit** button to return to the DBA Administration menu.

## Display index information

You can use the Display Index Information option to display sizing information for the database indexes. You can use this information to analyze the growth of the database indexes, which can indicate table growth.

- 1 On the DBA Administration menu, select the Display Index Information option.
- 2 Click the **Display** button. The Index Information screen is displayed.



SEGMENT NAME	TABLESPACE NAME	KBYTES	BLOCKS	EXTENTS	BYTES
DISTRO_ITEM_TO_UPLOAD_T	USERS	512	64	8	524288
DISTR_IDX1	USERS	64	8	1	65536
DI_PK	USERS	64	8	1	65536
DMS_LANGUAGE_MENU	DAT1	576	72	9	589824
DMS_LANGUAGE_MENU1	USERS	1024	128	16	1048576
DMS_LANGUAGE_MENU_LAN	IND2	256	32	4	262144
DMS_LANGUAGE_MENU_OP	IND2	704	88	11	720896
DMS_MENU	DAT3	3072	384	3	3145728
DMS_MENU1	USERS	512	64	8	524288
DMS_PICTURES	DAT4	576	72	9	589824
DMS_USER	DAT1	64	8	1	65536
DMS_USER1	USERS	512	64	8	524288
DOOR	DAT3	64	8	1	65536
DOOR1	USERS	512	64	8	524288
DOOR_BLOCK	DAT2	64	8	1	65536
DOOR_BLOCK1	USERS	512	64	8	524288
DOOR_LOCATION	IND1	64	8	1	65536
DUAL	SYSTEM	16	2	1	16384
DUC\$	SYSTEM	16	2	1	16384
DUMMY	DAT2	64	8	1	65536
ELIGIBLE_INVENTORY	DAT2	192	24	3	196608
ELIGIBLE_INVENTORY1	USERS	512	64	8	524288
ELIG_INV_ITEM_BF_OUQ	IND3	192	24	3	196608
EQUIPMENT_TYPE	DAT2	64	8	1	65536
EQUIPMENT_TYPE1	USERS	512	64	8	524288
ERROR\$	SYSTEM	296	37	6	303104

*Index Information Screen*

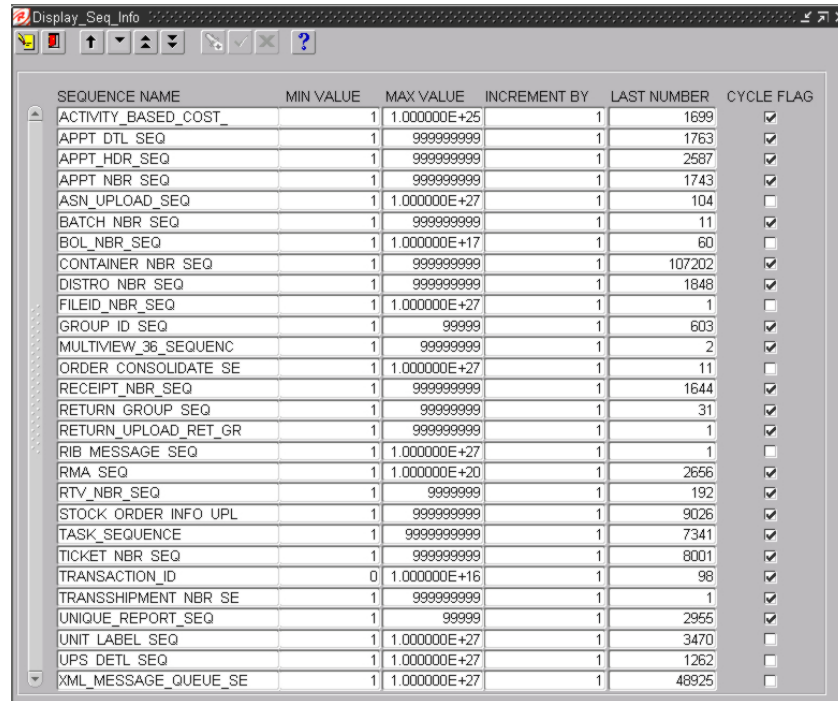
These fields are on the Index Information screen:

Field name	Field description
Target Index	Index name user wishes to query.
Segment Name	Name of index.
Tablespace Name	Tablespace name where the index resides.
Kbytes	Size of index in kilobytes.
Blocks	Number of blocks the index is using (1 block = 4096 bytes).
Extents	Current number of index extents.
Bytes	Size of index in bytes.

## Display sequences information

You can use the Display Sequence Information option to display sizing information specific to the sequences used by the system. You can use this information to determine whether a sequence is being called as many times as originally planned.

- 1 On the DBA Administration menu, select the Display Sequences Information option.
- 2 The Sequences Information screen is displayed, along with the sequence information already entered into the system.



SEQUENCE NAME	MIN VALUE	MAX VALUE	INCREMENT BY	LAST NUMBER	CYCLE FLAG
ACTIVITY_BASED_COST_	1	1.000000E+25	1	1699	<input checked="" type="checkbox"/>
APPT DTL SEQ	1	999999999	1	1763	<input checked="" type="checkbox"/>
APPT_HDR_SEQ	1	999999999	1	2587	<input checked="" type="checkbox"/>
APPT_NBR_SEQ	1	999999999	1	1743	<input checked="" type="checkbox"/>
ASN_UPLOAD_SEQ	1	1.000000E+27	1	104	<input type="checkbox"/>
BATCH_NBR_SEQ	1	999999999	1	11	<input checked="" type="checkbox"/>
BOL_NBR_SEQ	1	1.000000E+17	1	60	<input type="checkbox"/>
CONTAINER_NBR_SEQ	1	999999999	1	107202	<input checked="" type="checkbox"/>
DISTRO_NBR_SEQ	1	999999999	1	1848	<input checked="" type="checkbox"/>
FILEID_NBR_SEQ	1	1.000000E+27	1	1	<input type="checkbox"/>
GROUP_ID_SEQ	1	99999	1	603	<input checked="" type="checkbox"/>
MULTIVIEW_36_SEQUENC	1	999999999	1	2	<input checked="" type="checkbox"/>
ORDER_CONSOLIDATE_SE	1	1.000000E+27	1	11	<input type="checkbox"/>
RECEIPT_NBR_SEQ	1	999999999	1	1644	<input checked="" type="checkbox"/>
RETURN_GROUP_SEQ	1	999999999	1	31	<input checked="" type="checkbox"/>
RETURN_UPLOAD_RET_GR	1	999999999	1	1	<input checked="" type="checkbox"/>
RIB_MESSAGE_SEQ	1	1.000000E+27	1	1	<input type="checkbox"/>
RMA_SEQ	1	1.000000E+20	1	2656	<input checked="" type="checkbox"/>
RTV_NBR_SEQ	1	9999999	1	192	<input checked="" type="checkbox"/>
STOCK_ORDER_INFO_UPL	1	999999999	1	9026	<input checked="" type="checkbox"/>
TASK_SEQUENCE	1	999999999	1	7341	<input checked="" type="checkbox"/>
TICKET_NBR_SEQ	1	999999999	1	8001	<input checked="" type="checkbox"/>
TRANSACTION_ID	0	1.000000E+16	1	98	<input checked="" type="checkbox"/>
TRANSHIPMENT_NBR_SE	1	999999999	1	1	<input checked="" type="checkbox"/>
UNIQUE_REPORT_SEQ	1	99999	1	2955	<input checked="" type="checkbox"/>
UNIT_LABEL_SEQ	1	1.000000E+27	1	3470	<input type="checkbox"/>
UPS_DETL_SEQ	1	1.000000E+27	1	1262	<input type="checkbox"/>
XML_MESSAGE_QUEUE_SE	1	1.000000E+27	1	48925	<input type="checkbox"/>

***Sequences Information Screen***

These are the fields on the Sequences Information screen:

Field name	Field description
Sequence Name	Name of the database sequence.
Min Value	Minimum value of the sequence.
Max Value	Maximum value of the sequence.
Increment By	Increment, amount the sequence increases after each use.
Last Number	Last sequence value used. Some sequences cache the values in memory so this number does not increase until the cache is empty and a new group of numbers is cached into memory.

Field name	Field description
Cycle Flag	Cycle, whether the sequence rolls back to minimum value when the maximum value is reached.

- 3 Click the **Refresh** button to refresh the screen and view any new information.
- 4 Click the **Exit** button to return to the Main menu.

## Display the error log

Display Error Log option to display, view details, and delete logged errors. These are unanticipated errors or errors occurring in background processes. You can also print the error log.

- 1 On the Monitoring and Administration menu, select the Display Error Log option.
- 2 Click the **Display** button. The Error Log Inquiry screen is displayed, as shown in the following:

*Error Log Inquiry Screen*

These are the fields on the Error Log Inquiry screen:

Field name	Field description
Enter User, Code, Date	Enter any combination of user, error code, or error date to search for records.
User	User identification.
Error Time	The date and time the error was logged.
Code	The error code.
Source	Program where the error originated

- Click the **Display** button at the blank Enter User field to display a list of all existing errors.

**Note:** To display the errors for a particular user, enter the user name in the Enter User field.

- To display a specific error for a particular user, enter the user name in the Enter User field and the error code in the Code field.
- To display a specific error for a specific user for a particular date, enter the user name in the Enter User field, the error code in the Code field, and the date in the Date field.

Retek Distribution Management accepts any combination of the above fields.

## View error log details

- On the Monitoring and Administration menu, select the Display Error Log option, using the keypad arrow keys to move up and down the list.
- Click the **Display** button. The Error Log Inquiry screen is displayed.
- Enter the user, error code, or date that you want to view and click the **Display** button, or click the **Display** button at the blank Enter User field to display a list of all errors.
- Select the record you want to view in more detail, using the keypad arrow keys to move up and down the list, then click the **Details** button. The Error Log screen is displayed.

The screenshot shows a window titled "Error Detail" with a close button (X) in the top right corner. Inside the window, there are several input fields and labels:

- USER:** A text box containing "PAR3214".
- TIME:** A text box containing "03/05/2002 11:01:17".
- CODE:** A text box containing "40508".
- ERROR SOURCE:** A text box containing "ITEM\_MASTER\_EDITOR\_S".
- LOCATION ID:** A text box containing "Tr\_On\_Error".
- MESSAGE:** A text box containing "ORACLE error: unable to INSERT record."

At the bottom center of the window, there is a button labeled "Exit/Cancel".

*Error Log screen*



These are the fields on the Error Log screen:

Field name	Field description
User	Identification of the user who had the error.
Time	Date and time the error was logged.
Code	The error code.
Source	The program in which the error originated.
Location	The location within the source program where the error occurred.
Message	Full text of the error message.

- 5 Click the **Cancel** button to return to the Error Log screen.

## Delete error log records

- 1 On the Monitoring and Administration menu, select the Display Error Log Inquiry option, using the keypad arrow keys to move up and down the list.
- 2 Click the **Display** button to enter the Error Log Inquiry screen.
- 3 Enter the user, error code, or date that you want to delete and click the **Display** button, or click the **Display** button at the blank Enter User field to display a list of all errors.
- 4 Select the record you want to delete, using the keypad arrow keys to move up and down the list, then click the **Delete** button. A message box asks you to confirm the deletion. The message reads:  
Confirm Delete Operation (Yes/No)
- 5 Click on the Yes button to delete the error log record.

## Print the error log report

- 1 On the Monitoring and Administration menu, select the Display Error Log option, using the keypad arrow keys to move up and down the list.
- 2 Click the **Display** button to enter the Enter Log Inquiry screen.
- 3 Enter the user, error code or date that you want to print and click the **Display** button at the blank Enter User field to display the Error Log report, which lists all existing errors.
- 4 Click the **Print** button. The Report Destination Pop-Up screen is displayed.
- 5 Click the **Print** button to print the report.
- 6 Click the **Cancel** button to cancel printing the report. The Error Log report is shown.

03/20/02 Page 1

Error Log

USER ID	ERROR TIME	CODE	ERROR SOURCE	ERROR LOC	MESSAGE
ANDY	03-06 07:36	41067	REPRINT_NULL_LABELS_S	Tr_On_Error	Cannot find Menu Item: invalid ID.
ANDY	03-11 09:26	-4068	INV_INQ_BY_ITEM_S	Tr_On_Error	ORA-04068: existing state of packages () has been discarded ORA-04061: existing state of package body "PAR3214.AHL" has been invalidated
ANDYS	03-06 07:34	41067	REPRINT_NULL_LABELS_S	Tr_On_Error	Cannot find Menu Item: invalid ID.
ANDYS	03-06 07:34	41067	ERROR_LOG_S	Tr_On_Error	Cannot find Menu Item: invalid ID.
ANDYS	03-06 07:35	41067	USER_TABLE_EDITOR_S	Tr_On_Error	Cannot find Menu Item: invalid ID.
ANDYS	03-06 11:37	41067	ERROR_LOG_S	Tr_On_Error	Cannot find Menu Item: invalid ID.
ANDYS	03-06 14:51	41067	RETURN_CODE_EDITOR_S	Tr_On_Error	Cannot find Menu Item: invalid ID.
ANDYS	03-06 14:52	41067	REPRINT_NULL_LABELS_S	Tr_On_Error	Cannot find Menu Item: invalid ID.
LALIT	03-18 13:19	0	v_container_tote_id	before print order	DD
LALIT	03-18 13:19	0	PRINT ORDER	after report	DD

### *Error Log Report*

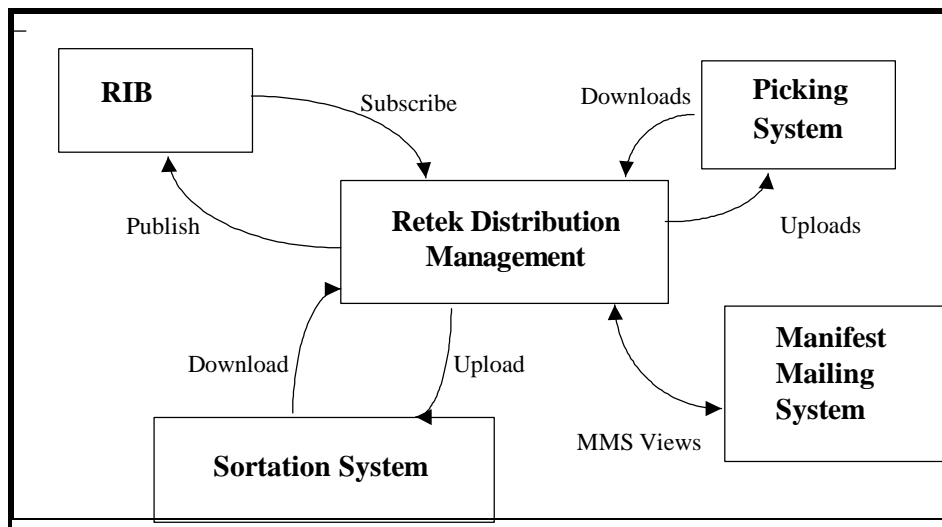
These are the fields on the Error Log report:

Field name	Field description
User ID	Identification of the user who had the error.
Error Time	Date and time the error was logged.
Code	The error code.
Error Source	Program where the error originated.
Error Loc	The location (procedure, block, etc.) within the source program where the error occurred.
Message	Full text of the error message.

## Chapter 3 - RDM RIB components

### Overview

This chapter and the next describe the various interfaces with the Retek Distribution Management System. These include Host System through the Retek Integration Bus (RIB), Picking System, Sortation System, and Manifest Mailing System links.



*Integration Points to RDM*

### Acronyms

These acronyms are used throughout this section:

**ASCII:** American National Standard Code for Information Interchange

**ASN:** Advance Shipment Notice

**DC:** Distribution Center

**PO:** Purchase Order

**SKU:** Stock Keeping Unit

## Terms

These terms are used throughout this section:

**Appointment:** A scheduled arrival of in-bound merchandise.

**ASN:** Advance Shipment Notice. A Host Download that provides either a list of containers and their contents, or a set of PO/Item/Destinations.

**Container:** A type of receptacle (such as a carton, pallet, tote, roll cage.) that contains items and/or other containers.

**Destination:** The ultimate source for containers. This covers out-bound destinations, including the DC itself and internal replenishment. Also referred to as the Shipping Destination. For consumer direct order fulfillment, this field is used to specify the shipment method or parcel carrier service.

**Download:** Any data file coming into Retek Distribution Management.

**Field:** An individual data element within a record.

**File:** The mechanism by which batch data is transferred. These are ASCII files.

**Future use:** The field is not currently used in Retek Distribution Management, but may be used in a future release.

**Host:** The controlling computer system. Often housed at corporate headquarters.

**Item:** A specified part number, SKU, etc.

**Optional:** The field is used for information purposes and is not required.

**Predistribution:** Allocation of merchandise in advance of receipt to facilitate flow through or cross-dock upon arrival, bypassing storage and going directly to break case picking area or shipping.

**Purchase Order:** The list of items and quantities authorized to receive from a specific vendor.

**Record:** A single line of data in a file.

**RIB:** Retek Integration Bus

**Upload:** Any data file going out from Retek Distribution Management to another system.

**Vendor:** A supplier of in-bound goods. Each PO is assigned to a vendor.

## Components

### Vendor subscription

Vendor messages are used by RDM to create and maintain Vendor and Vendor Address information. Vendor messages are published by a Host system.

Vendor Information is used by RDM in the inbound processing of Purchase Orders, Items, Receiving, Returns and RTV.

This family of messages is considered to be Foundation Data. Foundation Data indicates that the data is used as the basis for building other data models and is routed to every RDM installation in the enterprise.

### Vendor message structure

The Vendor family of messages can create, modify and delete Vendor records as well as create, modify, and delete Vendor Addresses. All of the message types are composed of the following sections:

- Message header—This is data about the Vendor including the Number and Name as well as auditing and sampling requirements for received product.
- Address record—Address Type (i.e. Billing, Shipping, etc) Primary Indicator and basic address information.

#### Message subscription process

The following is a description of the Vendor message subscription process:

- The RDM Vendor adapter recognizes that a message with the Vendor-specific name (i.e. VendorCre) exists on the RIB.
- The adapter calls the public PL/SQL procedure to “consume” the message. The public “consume” procedures are named:

RDMSUB\_VENDORCRE.CONSUME

RDMSUB\_VENDORMOD.CONSUME

RDMSUB\_VENDORDEL.CONSUME

RDMSUB\_VENDORADDRCRE.CONSUME

RDMSUB\_VENDORADDRMOD.CONSUME

RDMSUB\_VENDORADDRDEL.CONSUME

- The public procedure calls a set of generic consuming procedures and packages:

MESSAGE\_OBJECT

MESSAGEPRETABLE

MESSAGEPOSTTABLE

MESSAGEPOSTMESSAGE

### Message summary

All Vendor messages belong to the Vendor message family. The following table lists the single message that RDM subscribes to by its API name along with the document type definition (DTD) used for validation and parsing of the data during the message subscription process and the mapping documents that describes the data contained in the message.

Message Name (API)	Type (DTD)	Vendor Mapping Document
VendorCre	VendorDesc.dtd	Map_VendorDesc.xls
VendorMod	VendorHdrDesc.dtd	Map_VendorHdrDesc.xls
VendorDel	VendorRef.dtd	Map_VendorRef.xls
VendorAddrCre	VendorAddrDesc.dtd	Map_VendorAddrDesc.xls
VendorAddrMod	VendorAddrDesc.dtd	Map_VendorAddrDesc.xls
VendroAddrDel	VendorAddrRef.dtd	Map_VendorRef.xls

### PL/SQL procedures

This section describes the packages and procedures listed in the earlier “Message subscription process” section.

#### Public procedure

The RIB calls the public consume procedure that, in turn, passes the message to RDM’s Vendor adapter:

**RDMSUB\_VENDORCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Vendor create message.

**RDMSUB\_VENDORMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Vendor modify message.

**RDMSUB\_VENDORDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Vendor delete message.

**RDMSUB\_VENDORADDRCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Vendor Address create message.

**RDMSUB\_VENDORADDRMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Vendor Address modify message.

**RDMSUB\_VENDORADDRDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Vendor Address delete message.

**RDM generic processing**

The list of procedures and packages used for generic processing perform all of the parsing, additional processing, and validation against the database. For a more detailed view of the generic procedures and packages, see the RDM Generic Subscription Messages in this guide.

**Primary Vendor tables**

The following are the primary tables in RDM that hold Vendor data:

VENDOR

VENDOR\_ADDRESS

Detailed descriptions of these tables are in the RDM Data Model document.

**Location subscription**

Location messages, known as Ship Dest to RDM, are used by RDM to create and maintain Ship Destination records.

Location information is used by the warehouse to know where to ship merchandise and what methods / carriers to use.

This family of messages is considered to be Foundation Data. Foundation Data indicates that the data is used as the basis for building other data models and is routed to every RDM installation in the enterprise.

**Location message structure**

The Location or Ship Dest family of messages can create, modify and delete Ship Dest records. Ship Dest messages includes a Destination Identifier, address information, Carrier Information, Currency Codes, and Country Codes.

**Message subscription process**

The following is a description of the Location message subscription process:

- The RDM Location adapter recognizes that a message with a Location-specific name (i.e. LocationDesc) exists on the RIB.
- The adapter calls the public PL/SQL procedure to “consume” the message. The public “consume” procedures are named:

RDMSUB\_LOCCRE.CONSUME

RDMSUB\_LOCMOD.CONSUME

RDMSUB\_LOCDEL.CONSUME

- The public procedure calls a set of generic consuming procedures and packages:

MESSAGE\_OBJECT

MESSAGEPRETABLE

MESSAGEPOSTTABLE

MESSAGEPOSTMESSAGE

See the section “PL/SQL procedures” later in this overview for a summary of the procedures and functions involved in Location message processing.

### Message summary

All Location messages belong to the Location message family. The following table lists the single message that RDM subscribes to by its short name along with the document type definition (DTD) used for validation and parsing of the data during the message subscription process and the mapping documents that describes the data contained in the message.

Message Name (API)	Type (DTD)	Location Mapping Document
LOCCRE	LocationDesc.dtd	Map_StoreDesc.xls
LOCMOD	LocationDesc.dtd	Map_StoreDesc.xls
LOCDEL	LocationRef.dtd	Map_StoreRef.xls

## PL/SQL procedures

This section describes the procedures and functions listed in the earlier “Message subscription process” section.

### Public procedures

The RIB calls the public consume procedure that, in turn, passes the message to RDM’s Location adapter:

**RDMSUB\_LOCCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Location create message.

**RDMSUB\_LOCMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Location modify message.

**RDMSUB\_LOCDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Location delete message.

### RDM generic processing

The list of procedures and packages used for generic processing perform all of the parsing, additional processing, and validation against the database. For a more detailed view of the generic procedures and packages, see the RDM Generic Subscription Messages in this guide.

## Primary Location tables

The following are the primary tables in RDM that hold Location data:

- SHIP\_DEST

Detailed descriptions of these tables are in the RDM Data Model document.



## Item subscription

The Item messages are used by RDM to create and maintain Item and Item supporting information. Item messages are published by a Host system.

Items represent the actual merchandise that is received and shipped from the warehouse. The Item messages provide detail information about the merchandise including the Vendor, dimensions, and user defined attributes.

This family of messages is considered to be Foundation Data. Foundation Data indicates that the data is used as the basis for building other data models and is routed to every RDM installation in the enterprise.

## Item message structure

The Item family of messages consists of the following areas: Item, Supplier Information, Attributes, Differentiators, Bill Of Materials, and UPC. Each of these areas have create, modify and delete messages. A brief description of each node is provided below.

- Item—This is data about the Item itself including Vendor, Description, basic dimensions and weight. Also included in the Item node is the Item Differentiator information which provides a cross reference between the Item and the Differentiator / Differentiator Group tables.
- Item Supplier—The list of suppliers for list item including the primary supplier indicator.
- Item Supplier Country – The list of countries for each supplier including the primary country indicator. Additional information provide includes Inner Pack Size and TI / HI.
- Item Supplier Country Dimensions – The list of dimensions by object type (EACH, CARTON) by country.
- Item Attributes – The cross reference information between the Item and Attributes / Attribute Types.
- Bill of Materials – Information to relate the Master Item to the Component Items when creating pack items.
- Item UPC – Information to relate the Item to a UPC code.

### Message subscription process

The following is a description of the Item message subscription process:

- The RDM Item adapters recognize that a message with the Item-specific name (i.e. ITEMCre) exists on the RIB.
- The adapter calls the public PL/SQL procedure to “consume” the message. The public “consume” procedures are named:

RDMSUB\_ITEMCRE.CONSUME

RDMSUB\_ITEMHDRMOD.CONSUME

RDMSUB\_ITEMDEL.CONSUME

RDMSUB\_ITEMUPCCRE.CONSUME  
 RDMSUB\_ITEMUPCMOD.CONSUME  
 RDMSUB\_ITEMUPCDEL.CONSUME  
 RDMSUB\_ITEMBOMCRE.CONSUME  
 RDMSUB\_ITEMBOMMOD.CONSUME  
 RDMSUB\_ITEMBOMDEL.CONSUME  
 RDMSUB\_ITEMSUPCRE.CONSUME  
 RDMSUB\_ITEMSUPMOD.CONSUME  
 RDMSUB\_ITEMSUPDEL.CONSUME  
 RDMSUB\_ITEMSUPCTYCRE.CONSUME  
 RDMSUB\_ITEMSUPCTYCRE.CONSUME  
 RDMSUB\_ITEMSUPCTYCRE.COSUME  
 RDMSUB\_ITEMUDALOVCRE.CONSUME  
 RDMSUB\_ITEMUDALOVCRE.CONSUME  
 RDMSUB\_ITEMUDALOVCRE.CONSUME  
 RDMSUB\_ISCDIMCRE.CONSUME  
 RDMSUB\_ISCDIMCRE.CONSUME  
 RDMSUB\_ISCDIMCRE.CONSUME

- The public procedure calls a set of generic consuming procedures and packages:

MESSAGE\_OBJECT  
 MESSAGEPRETABLE  
 MESSAGEPOSTTABLE  
 MESSAGEPOSTMESSAGE

These procedures and packages perform all of the parsing, additional processing and validation against the database.

### Message summary

All Item messages belong to the Item message family. The following table lists the messages that RDM subscribes to by their short name along with the document type definition (DTD) used for validation and parsing of the data during the message subscription process and the mapping documents that describes the data contained in the messages.

Message Name (API)	Type (DTD)	Item Mapping Document
ITEMCRE	ItemDesc.dtd	Map_ItemDesc.xls
ITEMHDRMOD	ItemHdrDesc.dtd	Map_ItemHdrDesc.xls
ITEMDEL	ItemRef.dtd	Map_ItemRef.xls
ITEMBOMCRE	ItemBOMDesc.dtd	Map_ItemBOMDesc.xls
ITEMBOMMOD	ItemBOMDesc.dtd	Map_ItemBOMDesc.xls
ITEMBOMDEL	ItemBOMRef.dtd	Map_ItemBOMRef.xls
ITEMUPCCRE	ItemUPCDesc.dtd	ItemUPCDesc.dtd
ITEMUPCMOD	ItemUPCDesc.dtd	ItemUPCDesc.dtd
ITEMUPCDEL	ItemUPCRef.dtd	ItemUPCRef.dtd
ITEMUDALOVCRE	ItemUDALOVDesc.dtd	Map_ItemUDALOVDesc.xls
ITEMUDALOVMOD	ItemUDALOVDesc.dtd	Map_ItemUDALOVDesc.xls
ITEMUDALOVDEL	ItemUDALOVRef.dtd	Map_ItemUDALOVRef.xls
ITEMSUPCRE	ItemSupDesc.dtd	Map_ItemSupDesc.xls
ITEMSUPMOD	ItemSupDesc.dtd	Map_ItemSupDesc.xls
ITEMSUPDEL	ItemSupRef.dtd	Map_ItemSupRef.xls
ITEMSUPCTYCRE	ItemSupCtyDesc.dtd	Map_ItemSupCtyDesc.xls
ITEMSUPCTYMOD	ItemSupCtyDesc.dtd	Map_ItemSupCtyDesc.xls
ITEMSUPCTYDEL	ItemSupCtyRef.dtd	Map_ItemSupCtyRef.xls
ITEMISCDIMCRE	ISCDimDesc.dtd	Map_ISCDimDesc.xls
ITEMISCDIMMOD	ISCDimDesc.dtd	Map_ISCDimDesc.xls
ITEMISCDIMDEL	ISCDimRef.dtd	Map_ISCDimRef.xls

## PL/SQL procedures

This section describes the procedures and functions listed in the earlier “Message subscription process” section.

### Public procedure

The RIB calls the public consume procedure that, in turn, passes the message to RDM’s Item adapters:

**RDMSUB\_ITEMCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item create message.

**RDMSUB\_ITEMHDRMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item modify message.

**RDMSUB\_ITEMDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item delete message.

**RDMSUB\_ITEMBOMCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Bill of Materials create message.

**RDMSUB\_ITEMBOMMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Bill of Materials modify message.

**RDMSUB\_ITEMBOMDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Bill of Materials delete message.

**RDMSUB\_ITEMUPCCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item UPC create message.

**RDMSUB\_ITEMUPCMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item UPC modify message.

**RDMSUB\_ITEMUPCDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item UPC delete message.

**RDMSUB\_ITEMSUPCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item Supplier create message.

**RDMSUB\_ITEMSUPMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item Supplier modify message.

**RDMSUB\_ITEMSUPDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item Supplier delete message.

**RDMSUB\_ITEMSUPCTYCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item Supplier Country create message.

**RDMSUB\_ITEMSUPCTYMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item Supplier Country modify message.

**RDMSUB\_ITEMSUPCTYDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item Supplier Country delete message.

**RDMSUB\_ISCDIMCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item Supplier Country Dim create message.

**RDMSUB\_ISCDIMMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item Supplier Country Dim modify message.

**RDMSUB\_ISCDIMDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item Supplier Country Dim delete message.

**RDMSUB\_ITEMUDALOVCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item Attributes create message.

**RDMSUB\_ITEMUDALOVMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item Attributes modify message.

**RDMSUB\_ITEMUDALOVDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains an Item Attributes delete message.

### **RDM generic processing**

The list of procedures and packages used for generic processing perform all of the parsing, additional processing, and validation against the database. For a more detailed view of the generic procedures and packages, see the RDM Generic Subscription Messages in this guide.

## Primary Item tables

The following are the primary tables in RDM that hold Item data:

ITEM\_MASTER  
 ITEM\_SUPPLIER  
 ITEM\_SUPP\_COUNTRY  
 ITEM\_SUPP\_COUNTRY\_DIM  
 BILL\_OF\_MATERIALS  
 ITEM\_UPC  
 ITEM\_ATTRIBUTES

Detailed descriptions of these tables are in the RDM Data Model document.

## UDA subscription

UDA Information, known in RDM as Attributes and Attribute Types, is used to allow the user to define additional attributes for an Item. For example, for a cotton T-shirt, an Attribute of COTTON, meaning Cotton Short Sleeve T-Shirt, can be created and related to an Item through the Item Attribute table (see the ITEM subscription documentation for more information concerning the Item Attribute message).

This family of messages is considered to be Foundation Data. Foundation Data indicates that the data is used as the basis for building other data models and is routed to every RDM installation in the enterprise.

## UDA message structure

The UDA family of messages consists of two message types: UDA (Attribute Types) and UDA Values (Attributes). Both messages are single node structures.

### UDA Type

- This message includes the UDA Identifier and Description.

### UDA Value Type

- This message includes the UDA Value Identifier and Description and the UDA Identifier.

### Message subscription process

The following is a description of the UDA message subscription process:

- The RDM UDA adapter recognizes that a message with the UDA-specific name (i.e. UDACRE) exists on the RIB.
- The adapter calls the public PL/SQL procedure to “consume” the message. The public “consume” procedures are named:

RDMSUB\_UDACRE.CONSUME  
 RDMSUB\_UDAMOD.CONSUME  
 RDMSUB\_UDADEL.CONSUME

RDMSUB\_UDAVALCRE.CONSUME

RDMSUB\_UDAVALMOD.CONSUME

RDMSUB\_UDAVALDEL.CONSUME

- The public procedure calls a set of generic consuming procedures and packages:

MESSAGE\_OBJECT

MESSGEPRETABLE

MESSAGEPOSTTABLE

MESSAGEPOSTMESSAGE

These procedures and packages perform all of the parsing, additional processing and validation against the database.

### Message summary

All UDA messages belong to the UDA message family. The following table lists the single message that RDM subscribes to by its short name along with the document type definition (DTD) used for validation and parsing of the data during the message subscription process and the mapping documents that describes the data contained in the message.

Message Name (API)	Type (DTD)	UDA Mapping Document
UDACRE	UDADesc.dtd	Map_UDADesc.xls
UDAMOD	UDADesc.dtd	Map_UDADesc.xls
UDADEL	UDARef.dtd	Map_UDARef.xls
UDAVALCRE	UDAValDesc.dtd	Map_UDAValDesc.xls
UDAVALMOD	UDAValDesc.dtd	Map_UDAValDesc.xls
UDAVALDEL	UDAValDesc.dtd	Map_UDAValDesc.xls

## PL/SQL procedures

This section describes the procedures and functions listed in the earlier “Message subscription process” section.

### Public procedures

The RIB calls the public consume procedure that, in turn, passes the message to RDM’s UDA adapter:

**RDMSUB\_UDACRE.CONSUME**-This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a UDA create message.

**RDMSUB\_UDAMOD.CONSUME**-This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a UDA modify message.

**RDMSUB\_UDADEL.CONSUME**-This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a UDA delete message.

**RDMSUB\_UDAVALCRE.CONSUME**-This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a UDA detail create message.

**RDMSUB\_UDAVALMOD.CONSUME** -This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a UDA detail modify message.

**RDMSUB\_UDAVALDEL.CONSUME**-This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a UDA detail delete message. Private consume function

### RDM generic processing

The list of procedures and packages used for generic processing perform all of the parsing, additional processing, and validation against the database. For a more detailed view of the generic procedures and packages, see the RDM Generic Subscription Messages in this guide.

## Primary UDA tables

The following are the primary tables in RDM that hold UDA data:

ATTRIBUTES

ATTRIBUTE\_TYPE

Detailed descriptions of these tables are in the RDM Data Model document.

## Differentiator subscription

Differentiators and Differentiator Groups are created and then associated to Items through the Item Differentiator table (see the ITEM subscription documentation for more information concerning the Item Differentiator message). This information allows the user further characterize and group Items.

This family of messages is considered to be Foundation Data. Foundation Data indicates that the data is used as the basis for building other data models and is routed to every RDM installation in the enterprise.

## Differentiator message structure

The Differentiator family of messages consists of three message types: Differentiators, Differentiator Groups and Differentiator Group Details. All of these messages are single node structures.

Differentiator Type

- This message includes a Differentiator Identifier, Description and Type.

Differentiator Group Type

- This message includes a Differentiator Group Identifier, Group Description and Type.



### Differentiator Group Details Type

- This message includes the Differentiator Identifier, Differentiator Group Identifier, and Description.

### Message subscription process

The following is a description of the Differentiator message subscription process:

- The RDM Differentiator adapter recognizes that a message with a Diff-specific name (i.e. DiffCre) exists on the RIB.
- The adapter calls the public PL/SQL procedure to “consume” the message. The public “consume” procedures are named:

RDMSUB\_DIFFCRE.CONSUME

RDMSUB\_DIFFMOD.CONSUME

RDMSUB\_DIFFDEL.CONSUME

RDMSUB\_DIFFGRPCRE.CONSUME

RDMSUB\_DIFFGRPMOD.CONSUME

RDMSUB\_DIFFGRPDEL.CONSUME

RDMSUB\_DIFFGRPDTLCRE.CONSUME

RDMSUB\_DIFFGRPDTLMOD.CONSUME

RDMSUB\_DIFFGRPDTLDEL.CONSUME

- The public procedure calls a set of generic consuming procedures and packages:

MESSAGE\_OBJECT

MESSAGEPRETABLE

MESSAGEPOSTTABLE

MESSAGEPOSTMESSAGE

### Message summary

All Differentiator messages belong to the Differentiator message family. The following table lists the single message that RDM subscribes to by its short name along with the document type definition (DTD) used for validation and parsing of the data during the message subscription process and the mapping documents that describes the data contained in the message.

Message Name (API)	Type (DTD)	Differentiator Mapping Document
DIFFCRE	DiffDesc.dtd	Map_DiffDesc.xls
DIFFMOD	DiffDesc.dtd	Map_DiffDesc.xls
DIFFDEL	DiffRef.dtd	Map_DiffRef.xls
DIFFGRPCRE	DiffGrpHdrDesc.dtd	Map_DiffGrpHdrDesc.xls
DIFFGRPMOD	DiffGrpHdrDesc.dtd	Map_DiffGrpHdrDesc.xls
DIFFGRPDEL	DiffGrpRef.dtd	Map_DiffGrpRef.xls
DIFFGRPDTLCRE	DiffGrpDtlDesc.dtd	Map_DiffGrpDtlDesc.xls
DIFFGRPDTLMO D	DiffGrpDtlDesc.dtd	Map_DiffGrpDtlDesc.xls
DIFFGRPDTLDEL	DiffGrpDtlRef.dtd	Map_DiffGrpDtlRef.xls

### PL/SQL procedures

This section describes the procedures and functions listed in the earlier “Message subscription process” section.

#### Public procedures

The RIB calls the public consume procedure that, in turn, passes the message to RDM’s Differentiator adapter:

**RMSSUB\_ASNOUTCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a BOL create message with.

#### Private consume function

The public procedure calls one private consume function to begin RMS’s consumption of the message and its data:

**RDMSUB\_DIFFCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Differentiator create message.

**RDMSUB\_DIFFMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Differentiator modify message.

**RDMSUB\_DIFFDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Differentiator delete message.

**RDMSUB\_DIFFGRPCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Differentiator Group create message.

**RDMSUB\_DIFFGRPMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Differentiator Group modify message.

**RDMSUB\_DIFFGRPDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Differentiator Group delete message.

**RDMSUB\_DIFFGRPDTLCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Differentiator Group Detail create message.

**RDMSUB\_DIFFGRPDTLMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Differentiator Group Detail modify message.

**RDMSUB\_DIFFGRPDTLDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Differentiator Group Detail delete message.

### **RDM generic processing**

The list of procedures and packages used for generic processing perform all of the parsing, additional processing, and validation against the database. For a more detailed view of the generic procedures and packages, see the RDM Generic Subscription Messages in this guide.

## **Primary Differentiator tables**

The following are the primary tables in RDM that holds Differentiator data:

DIFFERENTIATOR

DIFFERENTIATOR\_GROUP

DIFF\_GROUP\_DETAIL

Detailed descriptions of these tables are in the RDM Data Model document.

## Purchase Order subscription

Purchase Order messages are used by RDM to create and maintain PO and PO Detail information. Purchase Order messages are published by a Host system.

Purchase Order messages authorize a warehouse to be able receive merchandise from a Vendor. These messages provide information to the warehouse about the amount of each item that can be received into the warehouse as well as acceptable date ranges for delivery.

The Purchase Order messages are specific to a particular warehouse instance and therefore contain routing information so that the bus can guarantee successful delivery of the message to the appropriate DC.

## Purchase Order message structure

The Purchase Order family of messages can create, modify and delete Purchase Order records as well as create, modify, and delete Purchase Order details. All of the message types are composed of the following sections:

- Message header—This is data about the entire PO including Number, Vendor, and Delivery Date information.
- PO Header Location – Routing information used to for routing purposes only.
- PO Detail record—Specific Item and quantity information.

### Message subscription process

The following is a description of the PO message subscription process:

- The RDM PO adapter recognizes that a message with a PO-specific name (i.e. PODesc) exists on the RIB.
- The adapter calls the public PL/SQL procedure to “consume” the message. The public “consume” procedures are named:

RDMSUB\_POPHYCRE.CONSUME

RDMSUB\_POPHYMOD.CONSUME

RDMSUB\_POPHYDEL.CONSUME

RDMSUB\_PODTLPHYCRE.CONSUME

RDMSUB\_PODTLPHYMOD.CONSUME

RDMSUB\_PODTLPHYDEL.CONSUME

- The public procedure calls a set of generic consuming procedures and packages:

MESSAGE\_OBJECT

MESSAGEPRETABLE

MESSAGEPOSTTABLE

MESSAGEPOSTMESSAGE

### Message summary

All PO messages belong to the PO message family. The following table lists the single message that RDM subscribes to by its short name along with the document type definition (DTD) used for validation and parsing of the data during the message subscription process and the mapping documents that describes the data contained in the message.

Message Name (API)	Type (DTD)	Purchase Order Mapping Document
POPhysCre	POPhyDesc.dtd	Map_ POPhyDesc.xls
POPhysMod	POPhyDesc.dtd	Map_ POPhyDesc.xls
POPhysDel	POPhyRef.dtd	Map_ POPhyRef.xls
PODtPhysCre	PODtPhyDesc.dtd	Map_ PODtPhyDesc.xls
PODtPhysMod	PODtPhyDesc.dtd	Map_ PODtPhyDesc.xls
PODtPhysDel	PODtPhyRef.dtd	Map_ POPhyRef.xls

### PL/SQL procedures

This section describes the procedures and functions listed in the earlier “Message subscription process” section.

#### Public procedures

The RIB calls the public consume procedure that, in turn, passes the message to RDM’s PO adapters:

**RDMSUB\_POPHYCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a PO create message.

**RDMSUB\_POPHYMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a PO modify message.

**RDMSUB\_POPHYDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a PO delete message.

**RDMSUB\_PODTLPHYCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a PO Detail create message.

**RDMSUB\_PODTLPHYMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a PO Detail modify message.

**RDMSUB\_PODTLPHYDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a PO Detail delete message.

**RDM generic processing**

The list of procedures and packages used for generic processing perform all of the parsing, additional processing, and validation against the database. For a more detailed view of the generic procedures and packages, see the RDM Generic Subscription Messages in this guide.

**Primary Purchase Order tables**

The following are the primary tables in RDM that hold PO data:

PO

PO\_DETAIL

Detailed descriptions of these tables are in the RDM Data Model document.

**Inbound Work Order subscription**

Inbound Work Order messages are used by RDM to create and maintain work order information. Inbound Work Order messages are published by a Host system.

Inbound Work Order messages represent a request for the warehouse to perform work on the merchandise before it is shipped to the stores or customers.

The Inbound Work Order messages are specific to a particular warehouse instance and therefore contain routing information so that the bus can guarantee successful delivery of the message to the appropriate DC.

**Inbound Work Order message structure**

The Inbound Work Order family of messages can create, modify and delete Inbound Work Order records. The message includes the following information: Item, Wip Code, Sequence and Instructions.

**Message subscription process**

The following is a description of the Inbound Work Order message subscription process:

- The RDM Inbound Work Order adapters recognize that a message with an Inbound Work Order-specific name (i.e. INBDWOCre) exists on the RIB.
- The adapter calls the public PL/SQL procedure to “consume” the message. The public “consume” procedures are named:

RDMSUB\_INBDWOCRE.CONSUME

RDMSUB\_INBDWOMOD.CONSUME

RDMSUB\_INBDWODEL.CONSUME

- The public procedure calls a set of generic consuming procedures and packages:

MESSAGE\_OBJECT

MESSGEPRETABLE

MESSAGEPOSTTABLE

MESSAGEPOSTMESSAGE

These procedures and packages perform all of the parsing, additional processing and validation against the database.

### Message summary

All Inbound Work Order messages belong to the Inbound Work Order message family. The following table lists the single message that RDM subscribes to by its short name along with the document type definition (DTD) used for validation and parsing of the data during the message subscription process and the mapping documents that describes the data contained in the message.

Message Name (API)	Type (DTD)	Inbound Work Order Mapping Document
INBDWOCre	WODesc.dtd	Map_ WODesc.xls
INBDWOMod	WODesc.dtd	Map_ WODesc.xls
INBDWODEl	WOREf.dtd	Map_ WOREf.xls

## PL/SQL procedures

This section describes the procedures and functions listed in the earlier “Message subscription process” section.

### Public procedure

The RIB calls the public consume procedure that, in turn, passes the message to RDM’s Inbound Work Order adapter:

**RDMSUB\_INBDWOCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Inbound Work Order create message.

**RDMSUB\_INBDWOMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Inbound Work Order modify message.

**RDMSUB\_INBDWODEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Inbound Work Order delete message.

**RDM generic processing**

The list of procedures and packages used for generic processing perform all of the parsing, additional processing, and validation against the database. For a more detailed view of the generic procedures and packages, see the RDM Generic Subscription Messages in this guide.

**Primary Inbound Work Order tables**

The following are the primary tables in RDM that hold Inbound Work Order data:

**INBOUND\_WORK\_ORDER**

Detailed descriptions of these tables are in the RDM Data Model document.

**Inbound ASN subscription**

Inbound ASN messages are used by RDM to create and maintain Advanced Shipment Information within the system. Inbound ASN messages are published by an outside Vendor or by another warehouse through the publication and transformation on an Outbound ASN.

Inbound ASN messages represent an Advanced Ship Notice of incoming merchandise. These messages provide information to the warehouse about the amount of each item that is coming to the DC.

The Inbound ASN messages are specific to a particular warehouse instance and therefore contain routing information so that the bus can guarantee successful delivery of the message to the appropriate DC.

**Inbound ASN message structure**

The Inbound ASN messages come in two style depending on the type. PO Type ASNs provide information about the Items being shipped to the warehouse. Carton Type ASNs also provide information about the Items and in addition supply all of the carton information as well. The two structures share common nodes which are detailed below:

- Message header—ASN Number, Type, Carrier.
- PO record—Purchase Order information.
- Carton—(for Carton Type ASNs) Container Identifier, dimensions
- Items— Details about all items in the Container.

**Message subscription process**

The following is a description of the BOL message subscription process:

- The RMS external ASN adapter recognizes that a message with the ASN-specific name (i.e. ASNInPOCre) exists on the RIB.



- The adapter calls the public PL/SQL procedure to “consume” the message. The public “consume” procedure is named:

RDMSUB\_ASNINPOCRE.CONSUME

RDMSUB\_ASNINPOMOD.CONSUME

RDMSUB\_ASNINPODEL.CONSUME

RDMSUB\_ASNINCTNCRE.CONSUME

RDMSUB\_ASNINCTNMOD.CONSUME

RDMSUB\_ASNINCTNDEL.CONSUME

- The public procedure calls a set of generic consuming procedures and packages:

MESSAGE\_OBJECT

MESSAGEPRETABLE

MESSAGEPOSTTABLE

MESSAGEPOSTMESSAGE

### Message summary

All ASN messages belong to the ASN message family. The following table lists the single message that RDM subscribes to by its short name along with the document type definition (DTD) used for validation and parsing of the data during the message subscription process and the mapping documents that describes the data contained in the message.

Message Short Name	Type (DTD)	Inbound ASN Mapping Document
ASNINPOCre	ASNInPODesc.dtd	Map_ ASNInPODesc.xls
ASNINPOMod	ASNInPODesc.dtd	Map_ ASNInPODesc.xls
ASNINPODel	ASNInPORef.dtd	Map_ ASNInPORef.xls
ASNINCTNCRE	ASNInCtnDesc.dtd	Map_ ASNInCtnDesc.xls
ASNINCTNCRE	ASNInCtnDesc.dtd	Map_ ASNInCtnDesc.xls
ASNINCTNCRE	ASNInCtnRef.dtd	Map_ ASNInCtnRef.xls

## PL/SQL procedures

This section describes the procedures and functions listed in the earlier “Message subscription process” section.

### Public procedures

The RIB calls the public consume procedure that, in turn, passes the message to RDM’s ASN adapter:

**RDMSUB\_ASNINPOCRE\_CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a ASN PO Type create message.

**RDMSUB\_ASNINPOMOD\_CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a ASN PO Type modify message.

**RDMSUB\_ASNINPODEL\_CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a ASN PO Type delete message.

**RDMSUB\_ASNINCTNCRE\_CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a ASN Container Type create message.

**RDMSUB\_ASNINCTNMOD\_CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a ASN Container Type modify message.

**RDMSUB\_ASNINCTNDEL\_CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a ASN Container Type delete message.

### RDM generic processing

The list of procedures and packages used for generic processing perform all of the parsing, additional processing, and validation against the database. For a more detailed view of the generic procedures and packages, see the RDM Generic Subscription Messages in this guide.

## Primary Inbound ASN tables

The following are the primary tables in RDM that hold ASN data:

ASN  
ASN\_ITEM  
CONTAINER  
CONTAINER\_ITEM  
PO  
PO\_DETAIL

Detailed descriptions of these tables are in the RDM Data Model document.

## Stock Order subscription

Stock Order messages are used by RDM to create and maintain stock order, stock allocation, and component ticketing information. Stock Order messages are published by a Host system.

Stock Order messages represent a request for merchandise to be sent to another location. These messages provide information to the warehouse about the amount of each item that needs to be processed and shipped to the provided destination along with billing and shipping address information.

The Stock Order messages are specific to a particular warehouse instance and therefore contain routing information so that the bus can guarantee successful delivery of the message to the appropriate DC.

## Stock Order message structure

The Stock Order family of messages can create, modify and delete Stock Order records as well as create, modify, and delete Stock Orders details, Stock Allocation and Component Ticketing. All of the message types are composed of the following sections:

- Message header—This is data about the Stock Order including billing and shipping information, picking dates, and cartonization information.
- Allocation record – Requested Items, Destinations, and quantities.
- Component Ticketing record – Master and Component Item relationships.

### Message subscription process

The following is a description of the Stock Order message subscription process:

- The RDM Stock Order adapter recognizes that a message with the Stock Order-specific name (i.e. SOCRE) exists on the RIB.
- The adapter calls the public PL/SQL procedure to “consume” the message. The public “consume” procedures are named:

RDMSUB\_SOCRE.CONSUME

RDMSUB\_SOMOD.CONSUME

RDMSUB\_SODEL.CONSUME

RDMSUB\_SODTLCRE.CONSUME

RDMSUB\_SODTLMOD.CONSUME

RDMSUB\_SODTLDEL.CONSUME

- The public procedure calls a set of generic consuming procedures and packages:

MESSAGE\_OBJECT

MESSAGEPRETABLE

MESSAGEPOSTTABLE

MESSAGEPOSTMESSAGE

These procedures and packages perform all of the parsing, additional processing and validation against the database.

### Message summary

All Stock Order messages belong to the Stock Order message family. The following table lists the single message that RDM subscribes to by its short name along with the document type definition (DTD) used for validation and parsing of the data during the message subscription process and the mapping documents that describes the data contained in the message.

Message Short Name	Type (DTD)	Stock Order Mapping Document
SOCRE	SODesc.dtd	Map_AllocDesc.xls
SOMOD	SOHdrDesc.dtd	Map_AllocHdrDesc.xls
SODEL	SORef.dtd	Map_AllocRef.xls
SODCRE	SODtlDesc.dtd	Map_AllocDtlDesc.xls
SODMOD	SODtlDesc.dtd	Map_AllocDtlDesc.xls
SODDEL	SODtlRef.dtd	Map_AllocDtlRef.xls

### PL/SQL procedures

This section describes the procedures and functions listed in the earlier “Message subscription process” section.

#### Public procedures

The RIB calls the public consume procedure that, in turn, passes the message to RDM’s Stock Order adapter:

**RDMSUB\_SOCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Stock Order create message.

**RDMSUB\_SOMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Stock Order modify message.

**RDMSUB\_SODEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Stock Order delete message.

**RDMSUB\_SODCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Stock Order Detail create message.

**RDMSUB\_SODMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Stock Order Detail modify message.

**RDMSUB\_SODDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Stock Order Detail delete message.

### **RDM Generic Processing**

The list of procedures and packages used for generic processing perform all of the parsing, additional processing, and validation against the database. For a more detailed view of the generic procedures and packages, see the RDM Generic Subscription Messages in this guide.

## **Primary Stock Order tables**

The following are the primary tables in RDM that hold Stock Order data:

STOCK\_ORDER  
STOCK\_ALLOCATION  
COMPONENT\_TICKETING  
STOCK\_ALLOCATION\_CID

Detailed descriptions of these tables are in the RDM Data Model document.

## **Outbound Work Order subscription**

Outbound Work Order messages are used by RDM to create and maintain work order information. Outbound Work Order messages are published by a Host system.

Outbound Work Order messages represent a request to the warehouse to perform work on the merchandise before it is shipped to the stores or customers.

The Outbound Work Order messages are specific to a particular warehouse instance and therefore contain routing information so that the bus can guarantee successful delivery of the message to the appropriate DC.

## **Outbound Work Order message structure**

The Outbound Work Order family of messages can create, modify and delete Outbound Work Order records. The message includes the following information: distro, destination, item, WIP sequence number, WIP code, personalization, instructions, order line number, and the auto complete flag.

### **Message subscription process**

The following is a description of the Outbound Work Order message subscription process:

- The RDM Outbound Work Order adapters recognize that a message with an Outbound Work Order-specific name (i.e. OUTBDWOCre) exists on the RIB.

- The adapter calls the public PL/SQL procedure to “consume” the message. The public “consume” procedures are named:

RDMSUB\_OUTBDWOCRE.CONSUME

RDMSUB\_OUTBDWOMOD.CONSUME

RDMSUB\_OUTBDWODEL.CONSUME

- The public procedure calls a set of generic consuming procedures and packages:

MESSAGE\_OBJECT

MESSGEPRETABLE

MESSAGEPOSTTABLE

MESSAGEPOSTMESSAGE

These procedures and packages perform all of the parsing, additional processing and validation against the database.

### Message summary

All Outbound Work Order messages belong to the Outbound Work Order message family. The following table lists the single message that RDM subscribes to by its short name along with the document type definition (DTD) used for validation and parsing of the data during the message subscription process and the mapping documents that describes the data contained in the message.

Message Short Name	Type (DTD)	Outbound Work Order Mapping Document
OUTBDWOCre	OutBdWODesc.dtd	Map_ OutBdWODesc.xls
OUTBDWOMod	OutBdWODesc.dtd	Map_ OutBdWODesc.xls
OUTBDWODEl	OutBdWOWRef.dtd	Map_ OutBdWOWRef.xls

## PL/SQL procedures

This section describes the procedures and functions listed in the earlier “Message subscription process” section.

### Public procedure

The RIB calls the public consume procedure that, in turn, passes the message to RDM’s Outbound Work Order adapter:

**RDMSUB\_OUTBDWOCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Outbound Work Order create message.

**RDMSUB\_OUTBDWOMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Outbound Work Order modify message.

**RDMSUB\_OUTBDWODEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Outbound Work Order delete message.

### RDM generic processing

The list of procedures and packages used for generic processing perform all of the parsing, additional processing, and validation against the database. For a more detailed view of the generic procedures and packages, see the RDM Generic Subscription Messages in this guide.

## Primary Outbound Work Order tables

The following descriptions are for the primary tables in RDM that hold Outbound Work Order data:

**OUTBOUND\_WORK\_ORDER**

Detailed descriptions of these tables are in the RDM Data Model document.

## Pending Returns subscription

Pending Return messages are used by RDM to create and maintain Pending Returns information. Pending Returns messages are published by a Host system.

Pending Returns messages represent a notification to the warehouse of merchandise that is being returned to the warehouse. These messages provide information to the warehouse about the amount of each item that is being returned.

The Pending Returns messages are specific to a particular warehouse instance and therefore contain routing information so that the bus can guarantee successful delivery of the message to the appropriate DC.

## Pending Returns message structure

The Pending Returns family of messages can create, modify and delete Pending Returns records as well as create, modify, and delete Pending Returns details. All of the message types are composed of the following sections:

- Message header—This is data about the RMA Number, PRO Number, Receipt Date.
- Detail record – The item and quantity.

### Message subscription process

The following is a description of the Vendor message subscription process:

- The RDM Pending Return adapters recognizes messages with Pending Returns-specific name (i.e. PendRetCre) exist on the RIB.
- The adapter calls the public PL/SQL procedure to “consume” the message. The public “consume” procedures are named:

RDMSUB\_PENDRETCRE.CONSUME

RDMSUB\_PENDRETMOD.CONSUME

RDMSUB\_PENDRETDEL.CONSUME

RDMSUB\_PENDRETDTLCRE.CONSUME

RDMSUB\_PENDRETDTLMOD.CONSUME

RDMSUB\_PENDRETDTLDEL.CONSUME

- The public procedure calls a set of generic consuming procedures and packages:

MESSAGE\_OBJECT

MESSGEPRETABLE

MESSAGEPOSTTABLE

MESSAGEPOSTMESSAGE



### Message summary

All Pending Return messages belong to the Pending Returns message family. The following table lists the single message that RMS subscribes to by its short name along with the document type definition (DTD) used for validation and parsing of the data during the message subscription process and the mapping documents that describes the data contained in the message.

Message Name (API)	Type (DTD)	Pending Returns Mapping Document
PendRetCre	PendRtrnDesc.dtd	Map_ PendRtrnDesc.xls
PendRetMod	PendRtrnDesc.dtd	Map_ PendRtrnDesc.xls
PendRetDel	PendRtrnRef.dtd	Map_ PendRtrnRef.xls
PendRetDtlCre	PendRtrnDtlDesc.dtd	Map_ PendRtrnDtlDesc.xls
PendRetDtlMod	PendRtrnDtlDesc.dtd	Map_ PendRtrnDtlDesc.xls
PendRetDtlDel	PendRtrnDtlRef.dtd	Map_ PendRtrnDtlRef.xls

### PL/SQL procedures

This section describes the procedures and functions listed in the earlier “Message subscription process” section.

#### Public procedure

The RIB calls the public consume procedure that, in turn, passes the message to RDM’s Pending Returns adapters:

**RDMSUB\_PENDRETCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Pending Returns create message.

**RDMSUB\_PENDRETMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Pending Returns modify message.

**RDMSUB\_PENDRETDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Pending Returns delete message.

**RDMSUB\_PENDRETDTLCRE.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Pending Returns Detail create message.

**RDMSUB\_PENDRETDTLMOD.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Pending Returns Detail modify message.

**RDMSUB\_PENDRETDTLDEL.CONSUME**—This procedure accepts a XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Pending Returns Detail delete message.

**RDM generic processing**

The list of procedures and packages used for generic processing perform all of the parsing, additional processing, and validation against the database. For a more detailed view of the generic procedures and packages, see the RDM Generic Subscription Messages in this guide.

**Primary Pending Returns tables**

The following are the primary tables in RDM that hold Pending Returns data:

PENDING\_RETURNS

PENDING\_RETURNS\_DETAIL

Detailed descriptions of these tables are in the RDM Data Model document.

**Inbound ASN publish**

RDM is responsible for communicating Inbound ASN Information to the Host System. Inbound ASN is defined as ASN Information originating in the RDM System. Inbound ASNs can be Container or PO Type ASNs. PO Type ASNs detail item information to be received at a unit level, not container level information. Container Type Inbound ASNs detail item information to be received at a container level. Container information includes Container ID, Destinations, Distro Number, Unit Quantity, PO and Item.

Inbound ASN messages are communicated to the Host once it has been Appointed. The entire hierarchical message is sent. To modify an ASN, the ASN must not be associated to an Appointment. Once modified, the entire hierarchical message is resent.

**Inbound ASN tables**

The RDM tables are populated when a record is recreated in RDM's screens.

**ASN\_TO\_UPLOAD:** Stores the XML messages that are to be transmitted by the BUS.

**ASN\_ITEM\_UPLOAD:** Stores the XML messages that are to be transmitted by the BUS.

**ASN\_CONT\_UPLOAD:** Stores the XML messages that are to be transmitted by the BUS.

**ASN\_PO\_UPLOAD:** Stores the XML messages that are to be transmitted by the BUS.

**INBOUND\_ASN\_QUEUE:** The Upload Table gives a complete snapshot of the data to be sent to the host system.

## ASN Inbound messages

There are two messages that pertain to Inbound ASN publishing.:

Message Short Name	Message Family Name
ASNInCre	ASNIn
ASNInDel	ASNIn

### Message family managers

This section describes the message family manager(s) (MFM) and the XML Builder Procedure for Inbound ASN

**RDMMFMM\_ASNIN** – This MFM retrieves messages from the message queue. It contains the public procedure GETNXT, which retrieves the next message on the queue.

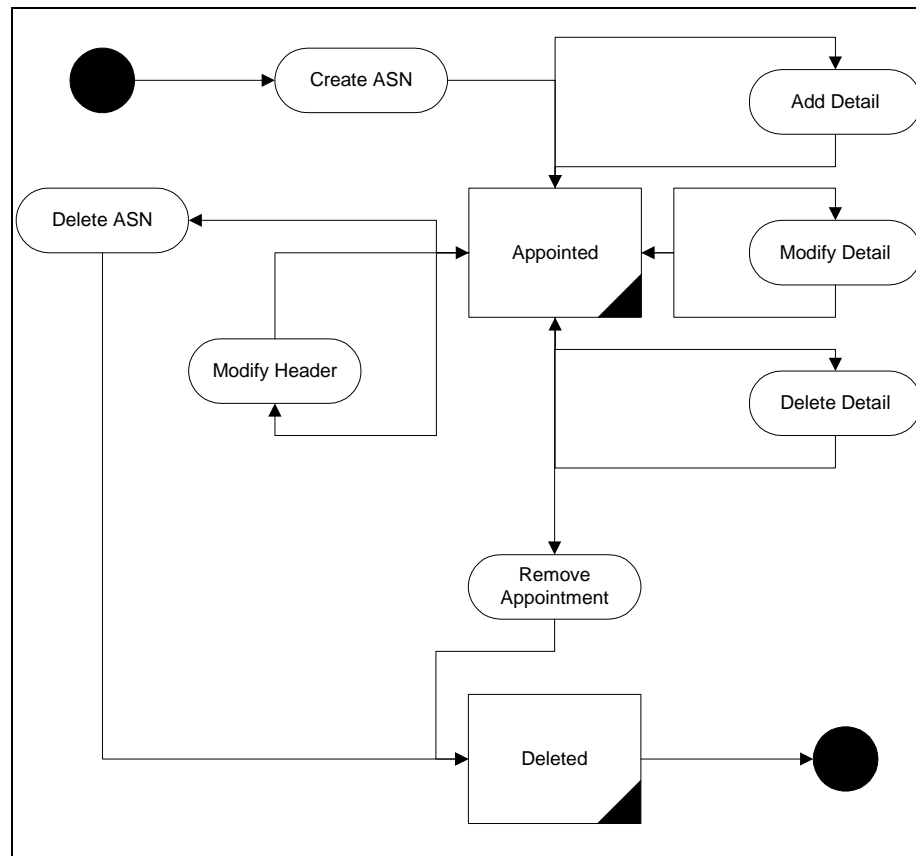
**INBOUND\_ASN\_BUILD\_XML** – Reads the Upload Table(s) and populates the Queue Table with their data.

### Message summary

The following table lists each message, by descriptive name and short name, along with the document type definition (DTD) used for validation during the message creation process, and the mapping document that describes the data contained in the message.

Message Description	Message Short Name	Type (DTD)	ASN Inbound Mapping Document
ASN IN Create	ASNInCre	ASNInDesc.dtd	Map_ASNInDesc.xls
ASN IN Delete	ASNInDel	ASNInRef.dtd	Map_ASNInRef.xls

## State diagram



## Description of activities

**Create Inbound ASN messages**

- 1 **Prerequisites:** Must be ASN appointment and a valid ASN.
- 2 **Activity Detail:** Assign the ASN to an Appointment.
- 3 **Messages:** When Inbound ASN Messages are created, the “Inbound ASN Create” data is inserted into the ASN\_Upload table. The Inbound ASN Create message is a hierarchical message containing a full snapshot of the Inbound ASN Message at the time the ASN was appointed.

**Delete Inbound ASN Messages**

- 1 **Prerequisites:** Must be ASN appointment and a valid ASN.
- 2 **Activity Detail:** Remove the ASN from the Appointment
- 3 **Messages:** When Inbound ASN Delete Messages are created, the “Inbound ASN Delete” data is inserted into the ASN\_Upload table. The Inbound ASN Create message is a hierarchical message containing a full snapshot of the Inbound ASN Message at the time the ASN was appointed.

## Triggers

None

## Message family manager procedures

### Public procedures

**GETNXT (O\_status\_code, O\_error\_msg, O\_message\_type, O\_message, I\_facility\_type, O\_from\_location, O\_ASN)** – This publicly exposed procedure is typically called by a RIB publication adaptor. Its parameters are well defined and arranged in a specific order. The message type is the RIB defined short message name, the message is the xml message, and the family keys are the key for the message as pertains to the family, not all of which will necessarily be populated for all message types. Status code is one of five values; these codes come from an EAI team defined RIB\_CODES package. For more discussion of the status codes, refer to the Error Handling Guidelines document.

The error text parameter contains application-generated information, such as the application's sequence number of the message that failed, and the Oracle or other error that occurred when the retrieval failed.

### Referenced stored procedures

**INBOUND\_ASN\_BUILD\_XML(I\_FACILITY\_TYPE)** – This procedure is responsible for retrieving the Inbound ASN Message Data from the ASN\_Upload / ASN\_Item\_Upload / ASN\_Cont\_Upload / ASN\_Item\_Upload tables, creating the appropriate XML, inserting the XML into the Inbound\_ASN\_Queue table, and marking the ASN\_Upload->Pub\_Status to 'S' and the ASN\_Upload->Transaction\_TS to the SYSDATE.

## Appointments/Receipts publish

RDM is responsible for communicating Appointment Information to the Host System. Appointment information consists of the Appointment Number, PO Information, Item Details, Scheduled Units and as well as ASN Information when related to an ASN.

Appointment messages are transmitted to the Host once the Appointment as been scheduled. Once scheduled, Appointment messages will be communicated at the addition, modification, or deletion of a detail, a modification of the header information such as arrival time, or at the Open, Close, and Deletion of the appointment.

RDM is responsible for communicating Receipt Information to the Host System.

Receipt information is at the container level. It is uploaded to the host from the container level or when an appointment is closed depending on an RDM system parameter. Receipt Info Upload will include appointment information, item number, ASN number if applicable, quantity, purchase order number, disposition changes, and type of receipt.

Receipt types include:

- Initial Receipt
- Adjustment to an already uploaded receipt

Both types of receipts contain the same information listed above.

## Receipt/Appointment tables

The RDM tables are populated when a record is created in RDM.

**APPT\_DETAIL\_TO\_UPLOAD:** Stores the XML messages that are to be transmitted by the BUS.

**APPT\_HEADER\_TO\_UPLOAD:** Stores the XML messages that are to be transmitted by the BUS.

**RECEIPT\_TO\_UPLOAD:** Stores the XML messages that are to be transmitted by the BUS.

**APPT\_RECEIPT\_QUEUE:** The Upload Table gives a complete snapshot of the data to be sent to the host system.

## Receipt/Appointment messages

There are four messages that pertain to Receipt/Appointment publishing:

Message Short Name	Message Family Name
ReceiptCre	Receiving
ReceiptMod	Receiving
AppointCre	Receiving
AppointHdrMod	Receiving
AppointDel	Receiving
AppointDtlCre	Receiving
AppointDtlMod	Receiving
AppointDtlDel	Receiving

### Message family managers

This section describes the message family manager(s) (MFM) and the XML Builder Procedure for Appointments/Receipts

**RDMMFMM\_RECEIVING** – This MFM retrieves messages from the message queue. It contains the public procedure GETNXT, which retrieves the next message on the queue.

**APPOINTMENT\_BUILD\_XML** – Reads the Upload Table(s) and populates the Queue Table with their data.

**RECEIPT\_BUILD\_XML** – Reads the Upload Table(s) and populates the Queue Table with their data.

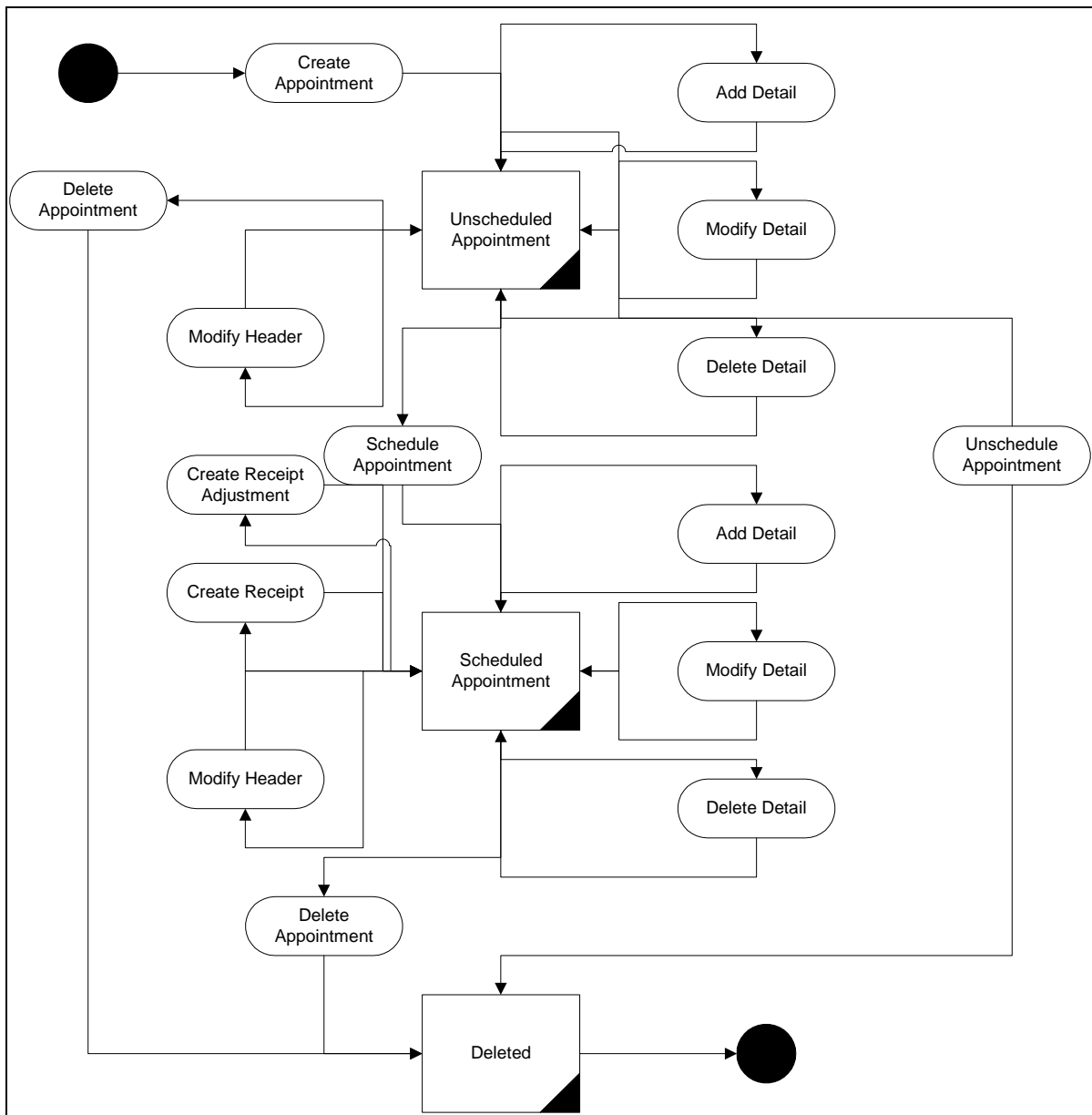
**Message summary**

The following table lists each message, by descriptive name and short name, along with the document type definition (DTD) used for validation during the message creation process, and the mapping document that describes the data contained in the message.

<b>Message Description</b>	<b>Message Short Name</b>	<b>Type (DTD)</b>	<b>Appointments/ Receipts Mapping Document</b>
Appointment Create	AppointCre	AppointDesc.dtd	Map_AppointDesc.xls
Appointment Modify	AppointMod	AppointHdrDesc.dtd	Map_AppointHdrDesc.xls
Appointment Delete	AppointDel	AppointRef.dtd	Map_AppointRef.xls
Appointment Detail Create	AppointDtlCre	AppointDtlDesc.dtd	Map_AppointDtlDesc.xls
Appointment Detail Modify	AppointDtlMod	AppointDtlDesc.dtd	Map_AppointDtlDesc.xls
Appointment Detail Delete	AppointDtlDel	AppointDtlRef.dtd	Map_AppointDtlRef.xls
Receipt Create	ReceiptCre	POReceiptDesc.dtd	Map_POReceiptDesc.xls
Receipt Modify	ReceiptMod	POReceiptDesc.dtd	Map_POReceiptDesc.xls



## State diagram



## Description of activities

**Appointment create**

- 1 **Prerequisites:** A valid door and trailer must exist to create an appointment.
- 2 **Activity Detail:** None
- 3 **Messages:** When Appointment Create Messages are created, the "Appointment Create" data is inserted into the Appt\_Header\_To\_Upload / Appt\_Detail\_To\_Upload table. The Appointment Create message is a hierarchical message containing a full snapshot of the Appointment Message at the time the first appointment detail record is added.

**Appointment modify**

- 1 **Prerequisites:** Appointment must exist.
- 2 **Activity Detail:** Change the Door, Appointment Time Stamp.
- 3 **Messages:** When Appointment Modify Messages are created, the “Appointment Modify” data is inserted into the Appt\_Header\_To\_Upload table. The Appointment Modify message is a flat message containing a full snapshot of the Appointment Modify Message at the time the appointment status is changed.

**Appointment delete**

- 1 **Prerequisites:** Appointment must exist and be in the appropriate status
- 2 **Activity Detail:** Cascade deletes to any associated detail tables.
- 3 **Messages:** When Appointment Delete Messages are created, the “Appointment Delete” data is inserted into the Appt\_Header\_To\_Upload table. The Appointment Delete message is a flat message containing the Appointment Number that was deleted.

**Appointment detail create**

- 1 **Prerequisites:** Valid appointment header and a valid PO and Item. If related to an ASN, the ASN must be valid.
- 2 **Activity Detail:** None
- 3 **Messages:** When Appointment Detail Create Messages are created, the “Appointment Detail Create” data is inserted into the Appt\_Header\_To\_Upload / Appt\_Detail\_To\_Upload table. The Appointment Detail Create message is a flat message containing a full snapshot of the Appointment Detail Create Message at the time the appointment detail is created.

**Appointment detail modify**

- 1 **Prerequisites:** Appointment detail record must exist in the appropriate status.
- 2 **Activity Detail:** Appropriate checks made to maintain data integrity.
- 3 **Messages:** When Appointment Detail Modify Messages are created, the “Appointment Detail Modify” data is inserted into the Appt\_Header\_To\_Upload / Appt\_Detail\_To\_Upload table. The Appointment Detail Modify message is a flat message containing a full snapshot of the Appointment Detail Modify Message at the time the appointment detail was modified changed.

**Appointment detail delete**

- 1 **Prerequisites:** Appointment detail record must exist in the appropriate status.
- 2 **Activity Detail:** None
- 3 **Messages:** When Appointment Detail Delete Messages are created, the “Appointment Detail Delete” data is inserted into the Appt\_Header\_To\_Upload / Appt\_Detail\_To\_Upload table. The Appointment Detail Delete message is a flat message containing a full snapshot of the Appointment Detail Delete Message at the time the appointment detail was created.

**Create receipt**

- 1 **Prerequisites:** Valid appointment must exist.
- 2 **Activity Detail:** Receipt of Container creates a Receipt to upload
- 3 **Messages:** When a receipt is created, the “Receipt Create” data is inserted into the Receipt\_To\_Upload table. The Receipt Create message is a flat message containing a full snapshot of the receipt at the time the receipt is created.

**Create receipt adjustment**

- 1 **Prerequisites:** Container must be received and the initial receipt upload must have been sent.
- 2 **Activity Detail:** Each container is individually checked using RDM functionality.
- 3 **Messages:** When a receipt adjustment is created, the “Receipt Adjustment” data is inserted into the Receipt\_To\_Upload table. The Receipt Adjustment message is a flat message containing a full snapshot of the receipt adjustment at the time the receipt adjustment is created.

**Triggers**

None

## Message family manager procedures

### Public procedures

**GETNXT (O\_status\_code, O\_error\_msg, O\_message\_type, O\_message, I\_facility\_type, O\_from\_location, O\_appt\_nbr)** – This publicly exposed procedure is typically called by a RIB publication adaptor. Its parameters are well defined and arranged in a specific order. The message type is the RIB defined short message name, the message is the xml message, and the family keys are the key for the message as pertains to the family, not all of which will necessarily be populated for all message types. Status code is one of five values; these codes come from an EAI team defined RIB\_CODES package. For more discussion of the status codes, refer to the Error Handling Guidelines document.

The error text parameter contains application-generated information, such as the application's sequence number of the message that failed, and the Oracle or other error that occurred when the retrieval failed.

### Referenced stored procedures

**APPOINTMENT\_BUILD\_XML(I\_FACILITY\_TYPE)** – This procedure is responsible for retrieving the Appointment Family Message Data from the Appt\_Header\_To\_Upload / Appt\_Detail\_To\_Upload tables, creating the appropriate XML, inserting the XML into the Appointment\_Queue table, and marking the Appt\_Header\_To\_Upload->Pub\_Status to 'S' and the Appt\_Header\_To\_Upload->Transaction\_TS to the SYSDATE. This procedure also calls the Receipt\_XML\_Builder procedure before processing Close messages. This done to process all receipts prior to closing the appointment.

**RECEIPT\_XML\_BUILDER(I\_FACILITY\_TYPE, I\_APPT\_NBR DEFAULT NULL)** – This procedure is responsible for retrieving the Receipt and Receipt Adjustment Information from the Receipt\_To\_Upload table, creating the appropriate XML, inserting the XML into the Receipt\_Queue table, and marking the Receipt\_To\_Upload->Pub\_Status to 'S' and the Receipt\_To\_Upload->Transaction\_TS to the SYSDATE.

In addition, this procedure determines when the Receipt/Receipt Adjustment records are created. RDM users have the ability to upload Receipt/Receipt Adjustments at a Container Level or at an Appointment Level. For Container Level, Receipt/Receipt Adjustments are sent as they are created. For Appointment Level, Receipt/Receipt Adjustments are sent once the Appointment has been received.

## Stock Order Status publish

RDM is responsible for communicating Stock Order status Information to the Host System. RDM will generate stock order status information upon detection of any changes to a stock order.

These statuses include:

- Successful Insert
- Successful Delete
- Store Reassign
- Detail Selected
- Detail Unselected
- Pick Created
- Pick Deleted
- Return to Stock
- Cartonization Complete
- Cartonization Reversed
- Expired Stock Order
- No Inventory

Information includes distro number, distro type, item information and quantities, and status.

## Stock Order Status tables

The RDM tables are populated when a record is updated in RDM.

**STOCK\_ORDER\_INFO\_QUEUE:** Stores the XML messages that are to be transmitted by the BUS.

**STOCK\_ORDER\_INFO\_UPLOAD:** The Upload Table gives a complete snapshot of the data to be sent to the host system.

## Stock Order Status messages

There is one messages that pertains to stock order status publishing:

Message Short Name	Message Family Name
SOStatusCre	Stock Orders

### Message family managers

This section describes the message family manager(s) (MFM) and the XML Builder Procedure for Stock Order Status:

**RDMMFM\_SOSTATUS** –This MFM retrieves messages from the message queue. It contains the public procedure GETNXT, which retrieves the next message on the message queue.

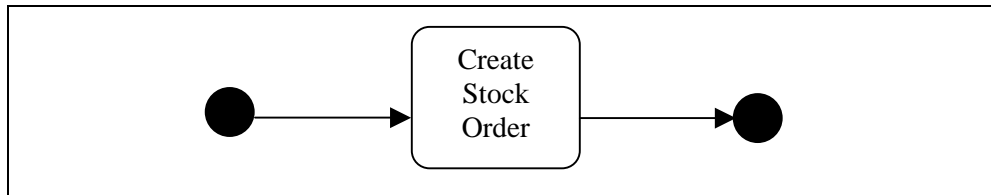
**SOS\_BUILD\_XML** –Reads from the Upload Table(s) and populates the Queue Table with their data.

### Message summary

The following table lists each message, by descriptive name and short name, along with the document type definition (DTD) used for validation during the message creation process, and the mapping document that describes the data contained in the message.

Message Description	Message Short Name	Type (DTD)	Stock Order Status Mapping Document
Store Create	SOSStatuxCre	SOSStatusDesc.dtd	Map_SOSStatusDesc.xls

### State diagram



### Description of activities

#### Create stock order info messages

- 1 **Prerequisites:** Valid distro number.
- 2 **Activity Detail:** Generate throughout the system per normal use of the system.
- 3 **Messages:** When Stock Order Info Messages are created, the “Stock Order Info Create” data is inserted into the Stock\_Order\_Info\_Upload table. The Stock Order Info Create message is a flat message containing a full snapshot of the Stock Order Info Messages at the time the inventory was affected.

### Triggers

None

## Message family manager procedures

### Public procedures

**GETNXT (O\_status\_code, O\_error\_msg, O\_message\_type, O\_message, I\_facility\_type)** – This publicly exposed procedure is typically called by a RIB publication adaptor. Its parameters are well defined and arranged in a specific order. The message type is the RIB defined short message name, the message is the xml message, and the family keys are the key for the message as pertains to the family, not all of which will necessarily be populated for all message types. Status code is one of five values; these codes come from an EAI team defined RIB\_CODES package. For more discussion of the status codes, refer to the Error Handling Guidelines document.

The error text parameter contains application-generated information, such as the application's sequence number of the message that failed, and the Oracle or other error that occurred when the retrieval failed.

### Referenced stored procedures

**SOS\_XML\_BUILDER(I\_FACILITY\_TYPE)** – This procedure is responsible for retrieving the Stock Order Info Message Data from the Stock\_Order\_Info\_Upload table, creating the appropriate XML, inserting the XML into the Stock\_Order\_Info\_Queue table, and marking the Stock\_Order\_Info\_Upload->Pub\_Status to 'S' and the Stock\_Order\_Info\_Upload->Transaction\_TS to the SYSDATE.

## Outbound ASN publish

RDM is responsible for communicating Outbound ASN Information to the Host System.

Outbound ASN Information consists of ASN Information, BOL Number, Manifest Information including Trailer and Carrier, Container Information including Items, Unit Quantities, Container ID, Destination and Distro Information.

An outbound ASN is generated for a distinct Shipping Trailer/Destination.

## Outbound ASN tables

The RDM tables are populated when a record is recreated in RDM's screens.

**BOL\_TO\_UPLOAD:** Stores the XML messages that are to be transmitted by the BUS.

**OUTBOUND\_ASN\_QUEUE:** The Upload Table gives a complete snapshot of the data to be sent to the host system.

## Outbound ASN messages

There is one message that pertains to Outbound ASN publishing.:

Message Short Name	Message Family Name
ASNOutCre	ASNOut

### Message family managers

This section describes the message family manager(s) (MFM) and the XML Builder Procedure for Outbound ASN

**RDMMFMM\_ASNOUT** – This MFM retrieves messages from the message queue. It contains the public procedure GETNXT, which retrieves the next message on the queue.

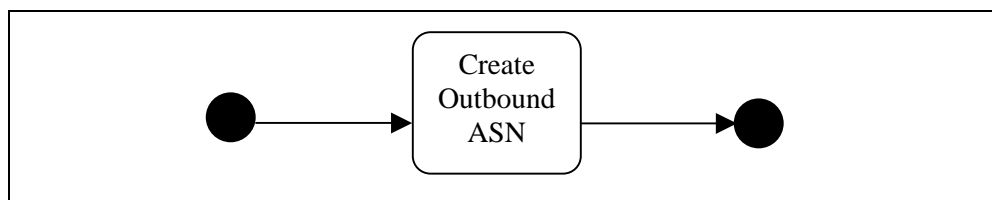
**OUTBOUND\_ASN\_BUILD\_XML** – Reads the Upload Table(s) and populates the Queue Table with their data.

### Message summary

The following table lists each message, by descriptive name and short name, along with the document type definition (DTD) used for validation during the message creation process, and the mapping document that describes the data contained in the message.

Message Description	Message Short Name	Type (DTD)	Outbound ASN Mapping Document
Outbund ASN Create	ASNOutCre	ASNOutDesc.dtd	Map_ASNOutDesc.xls

## State diagram



## Description of activities

### Create outbound ASN messages

- 1 **Prerequisites:** Trailer must be in a Shipped Status.
- 2 **Activity Detail:** None
- 3 **Messages:** When Outbound ASN Messages are created, the “Outbound ASN Create” data is inserted into the BOL\_To\_Upload table. The Outbound ASN Create message is a hierarchical message containing a full snapshot of the Outbound ASN Message at the time the shipment was created.



## Triggers

None

## Message family manager procedures

### Public procedures

**GETNXT (O\_status\_code, O\_error\_msg, O\_message\_type, O\_message, I\_facility\_type)** – This publicly exposed procedure is typically called by a RIB publication adaptor. Its parameters are well defined and arranged in a specific order. The message type is the RIB defined short message name, the message is the xml message, and the family keys are the key for the message as pertains to the family, not all of which will necessarily be populated for all message types. Status code is one of five values; these codes come from an EAI team defined RIB\_CODES package. For more discussion of the status codes, refer to the Error Handling Guidelines document.

The error text parameter contains application-generated information, such as the application's sequence number of the message that failed, and the Oracle or other error that occurred when the retrieval failed.

### Referenced stored procedures

**OUTBOUND\_ASN\_BUILD\_XML(I\_FACILITY\_TYPE)** – This procedure is responsible for retrieving the Outbound ASN Message Data from the BOL\_To\_Upload table and selecting from the Stock\_Order, Container, Container\_Item, and Manifest tables, creating the appropriate XML, inserting the XML into the Outbound\_ASN\_Queue table, and marking the BOL\_To\_Upload->Pub\_Status to 'S' and the BOL\_To\_Upload->Transaction\_TS to the SYSDATE.

## Inventory Adjustments publish

RDM is responsible for communicating Inventory Adjustments Information to the Host System.

Inventory Adjustments can be categorized as true inventory adjustments or inventory disposition changes.

True inventory adjustments are adjusting the actual quantity of the inventory available. Inventory disposition is changing the status of the inventory (i.e. from unavailable to sell to available to sell). True inventory adjustments must always have a disposition change, however, you may have an inventory disposition without a true inventory adjustment.

Inventory Disposition statuses include:

- Receipt in Process (RIP)
- Available to Sell (ATS)
- Pending WIP on Inventory (WIP code will be included)
- Trouble (Trouble code will be included)
- Distributed

The user can define alternate statuses to be uploaded to the host through an RDM defined editor

## Inventory Adjustments tables

The RDM tables are populated when a record is recreated in RDM's screens.

**INV\_ADJUSTMENT\_TO\_UPLOAD:** Stores the XML messages that are to be transmitted by the BUS.

**INVENTORY\_ADJ\_QUEUE\_QUEUE:** The Upload Table gives a complete snapshot of the data to be sent to the host system.

## Inventory Adjustments messages

There is one message that pertains to Inventory Adjustments publishing.:

Message Short Name	Message Family Name
InvAdjustCre	InvAdjust

### Message family managers

This section describes the message family manager(s) (MFM) and the XML Builder Procedure for Inventory Adjustments:

**RDMMFMM\_INVADJUST** – This MFM retrieves messages from the message queue. It contains the public procedure GETNXT, which retrieves the next message on the queue.

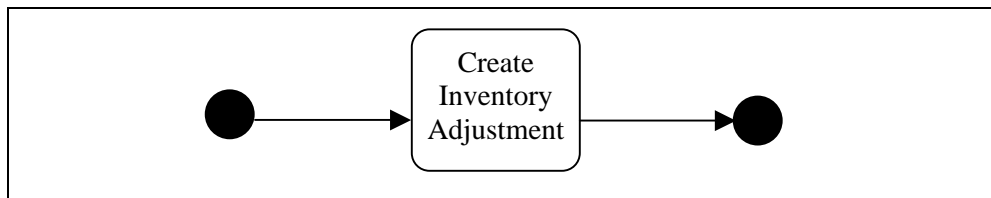
**INVADJ\_BUILD\_XML** – Reads the Upload Table(s) and populates the Queue Table with their data.

### Message summary

The following table lists each message, by descriptive name and short name, along with the document type definition (DTD) used for validation during the message creation process, and the mapping document that describes the data contained in the message.

Message Description	Message Short Name	Type (DTD)	Inventory Adjustments Mapping Document
Inventory Adjustments Create	InvAdjustCre	InvAdjustDesc.dtd	Map_InvAdjustDesc.xls

### State diagram



### Description of activities

#### Create inventory adjustments

- 1 **Prerequisites:** None.
- 2 **Activity Detail:** Inventory adjustments are created throughout the entire system as a result of normal processing.
- 3 **Messages:** When a Inventory Adjustments is created, the “Inventory Adjustments Create” data is inserted into the Inv\_Adjustment\_To\_Upload table. The Inventory Adjustments Create message is a flat message containing a full snapshot of the Inventory Adjustments at the time the Inventory Adjustments is created.

### Triggers

None

## Message family manager procedures

### Public procedures

**GETNXT (O\_status\_code, O\_error\_msg, O\_message\_type, O\_message, I\_facility\_type)** – This publicly exposed procedure is typically called by a RIB publication adaptor. Its parameters are well defined and arranged in a specific order. The message type is the RIB defined short message name, the message is the xml message, and the family keys are the key for the message as pertains to the family, not all of which will necessarily be populated for all message types. Status code is one of five values; these codes come from an EAI team defined RIB\_CODES package. For more discussion of the status codes, refer to the Error Handling Guidelines document.

The error text parameter contains application-generated information, such as the application's sequence number of the message that failed, and the Oracle or other error that occurred when the retrieval failed.

### Referenced stored procedures

**INVADJ\_XML\_BUILDER(I\_FACILITY\_TYPE)** – This procedure is responsible for retrieving the Inventory Adjustments Information from the InvAdj\_To\_Upload table, creating the appropriate XML, inserting the XML into the Inventory\_Adj\_Info\_Queue table, and marking the Inv\_Adjustment\_To\_Upload->Pub\_Status to 'S' and the Inv\_Adjustment\_To\_Upload->Transaction\_TS to the SYSDATE.

## Inventory Balance upload

When requested, Retek Distribution Management uploads an image of the current inventory. The format of the inventory balance record is as follows:

Field Description	Template	Description
Location (DC)	X (10)	Destination ID of the DC.
Transaction Date/Time	YYYYMMDDHH MI	Date of run.
Item ID	X (25)	Item identifier.
Available Units	sN (8) v N (4)	Units available for distribution.
Distributed Units	N (8) v N (4)	Units distributed includes: <ul style="list-style-type: none"> <li>• Units distributed but not yet picked.</li> <li>• Units picked but not yet manifested.</li> <li>• Units manifested but not yet shipped.</li> </ul>
Received Units	N (8) v N (4)	Units received but not put away.
Total Units	N (8) v N (4)	Sum of all units that physically exist: container status of: I, D, M, R, T, X.
Available Weight	N (8) v N (4)	Weight available for distribution of catch weight items.
Distributed Weight	N (8) v N (4)	Weight distributed includes: <ul style="list-style-type: none"> <li>• Weight distributed but not yet picked.</li> <li>• Weight picked but not yet manifested.</li> <li>• Weight manifested but not yet shipped.</li> </ul> Values only for catch weight items.
Received Weight	N (8) v N (4)	Weight received but not putaway for catch weight items.
Total Weight	N (8) v N (4)	Sum of all weight that physically exist: container status of: I, D, M, R, T, X. For catch weight items.

## Customer Returns publish

RDM is responsible for communicating Customer Returns Information to the Host System.

RDM provides the capability to process item level return information. Information to the host upon completion of the process will include: item information, unit quantity information, the RMA number, zero or more reason codes, zero or more action codes, and possibly replacement items and replacement quantities.

### Customer Returns tables

The RDM tables are populated when a record is created in RDM.

**RETURNS\_TO\_UPLOAD:** Stores the XML messages that are to be transmitted by the BUS.

**CUSTOMER\_RETURNS\_QUEUE:** The Upload Table gives a complete snapshot of the data to be sent to the host system.

## Customer Returns messages

There is one message that pertains to Customer Returns publishing:

Message Short Name	Message Family Name
CoRetCre	CustRet

### Message family managers

This section describes the message family manager(s) (MFM) and the XML Builder Procedure for RTV

**RDMFM\_CUSTRETURN**– This MFM retrieves messages from the message queue. It contains the public procedure GETNXT, which retrieves the next message on the queue.

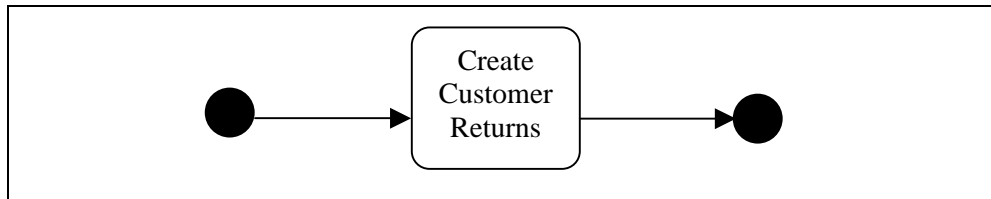
**CUSTOMER\_RETURN\_BUILD\_XML** – Reads the Upload Table(s) and populates the Queue Table with their data.

### Message summary

The following table lists each message, by descriptive name and short name, along with the document type definition (DTD) used for validation during the message creation process, and the mapping document that describes the data contained in the message.

Message Description	Message Short Name	Type (DTD)	Customer Returns Mapping Document
Customer Returns Create	CustRetCre	CustRetDesc.dtd	Map_CustRetDesc.xls

## State diagram



## Description of activities

**Create customer returns**

- 1 **Prerequisites:** There are no prerequisites for Customer Returns.
- 2 **Activity Detail:** There are no activity details, once the message has been processed there are no modifications.
- 3 **Messages:** When a Customer Return is created, the “Customer Returns Create” data is inserted into the Returns\_Upload table. The Customer Returns Create message is a flat message containing a full snapshot of the Customer Returns at the time the Customer Returns is created.

## Triggers

None

## Message family manager procedures

**Public procedures**

**GETNXT (O\_status\_code, O\_error\_msg, O\_message\_type, O\_message, I\_facility\_type)** – This publicly exposed procedure is typically called by a RIB publication adaptor. Its parameters are well defined and arranged in a specific order. The message type is the RIB defined short message name, the message is the xml message, and the family keys are the key for the message as pertains to the family, not all of which will necessarily be populated for all message types. Status code is one of five values; these codes come from an EAI team defined RIB\_CODES package. For more discussion of the status codes, refer to the Error Handling Guidelines document.

The error text parameter contains application-generated information, such as the application’s sequence number of the message that failed, and the Oracle or other error that occurred when the retrieval failed.

**Referenced stored procedures**

**CUSTOMER\_RETURN\_BUILD\_XML (I\_FACILITY\_TYPE)** – This procedure is responsible for retrieving the Customer Returns Information from the Returns\_Upload table, creating the appropriate XML, inserting the XML into the Customer\_Returns\_Queue table, and marking the Returns\_Upload->Pub\_Status to ‘S’ and the Returns\_Upload->Transaction\_TS to the SYSDATE.

## Return to Vendor publish

RDM is responsible for communicating RTV Information to the Host System. RTV information is sent to the Host when the DC chooses to return merchandise to the Vendor. Information includes Return Authorization Numbers, Vendor Information including address, Item and Quantity Information and Inventory Disposition Statuses.

## RTV tables

The RDM tables are populated when a record is posted in RDM.

**INV\_ADJUSTMENT\_TO\_UPLOAD:** Stores the XML messages that are to be transmitted by the BUS.

**RTV\_QUEUE:** The Upload Table gives a complete snapshot of the data to be sent to the host system.

## RTV messages

There is one message that pertains to RTV publishing.:

Message Short Name	Message Family Name
RTVCre	RTV

### Message family managers

This section describes the message family manager(s) (MFM) and the XML Builder Procedure for RTV

**RDMMFMM\_RTV** – This MFM retrieves messages from the message queue. It contains the public procedure GETNXT, which retrieves the next message on the queue.

**RTV\_BUILD\_XML** – Reads the Upload Table(s) and populates the Queue Table with their data.

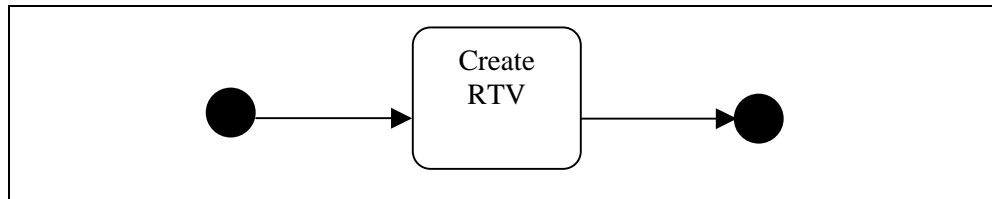
### Message summary

The following table lists each message, by descriptive name and short name, along with the document type definition (DTD) used for validation during the message creation process, and the mapping document that describes the data contained in the message.

Message Description	Message Short Name	Type (DTD)	Return to Vendor Mapping Document
RTV Create	RTVCre	RTVDesc.dtd	Map_RTVDesc.xls



## State diagram



## Description of activities

**Create RTV messages**

- 1 **Prerequisites:** Container must be in the appropriate status.
- 2 **Activity Detail:** All pending WIPs and Troubles are cleared prior to RTV.
- 3 **Messages:** When RTV Messages are created, the “RTV Create” data is inserted into the Stock\_Order\_Info\_Upload table. The RTV Create message is a flat message containing a full snapshot of the RTV Messages at the time the inventory was affected.

## Triggers

None

## Message family manager procedures

**Public procedures**

**GETNXT (O\_status\_code, O\_error\_msg, O\_message\_type, O\_message, I\_facility\_type)** – This publicly exposed procedure is typically called by a RIB publication adaptor. Its parameters are well defined and arranged in a specific order. The message type is the RIB defined short message name, the message is the xml message, and the family keys are the key for the message as pertains to the family, not all of which will necessarily be populated for all message types. Status code is one of five values; these codes come from an EAI team defined RIB\_CODES package. For more discussion of the status codes, refer to the Error Handling Guidelines document.

The error text parameter contains application-generated information, such as the application’s sequence number of the message that failed, and the Oracle or other error that occurred when the retrieval failed.

**Referenced stored procedures**

**RTV\_BUILD\_XML(I\_FACILITY\_TYPE)** – This procedure is responsible for retrieving the RTV Message Data from the Inv\_Adjustment\_To\_Upload table where the adjustment\_reason\_code = 90, creating the appropriate XML, inserting the XML into the RTV\_Queue table, and marking the Inv\_Adjustment\_To\_Upload->Pub\_Status to ‘S’ and the Inv\_Adjustment\_To\_Upload->Transaction\_TS to the SYSDATE.

## Streamsoft components

### Space Locations

RDM is responsible for communicating Forward Unit and Forward Case Picking Location (FPL and FCPL respectively) information to a third-party SKU profiling system for the purposes of warehouse optimization.

RDM FPL/FCPL information can be published in one of two ways. The first is through a Distribution Center (DC) Profiling support function provided within RDM. The second is through a series of location related event triggers that result in the location data being sent. These event triggers include:

- Creation or Deletion of a new unit or case picking (a.k.a. published) location.
- Updates to a published location's type, zone, status, put-away sequence, or pick sequence.
- Deletion or Update of information pertaining to a published location's type. Such information includes description, container capacity, length, width, height, max standard units, volume type, unit cost, and whether or not the location is for unit or case picking.
- Update of information pertaining to a published location's zone, such as description, pick priority, region or work area.
- Insert, Delete, or Update of an item to/from a picking location when the item has been SKU optimized and is assigned for SKU publishing. In this case, the location information for the picking location where the item has been assigned/unassigned is sent.

The information sent to the third-party system on an add or modify will include: location id, zone information, items assigned to that location for picking, and location type information such as whether the location is unit or case pick, length, height, etc. Deletes of location information will include only the DC Destination ID (from location) and the location id for the location being deleted.

### Space Location tables

The following tables are populated when the location information is published by RDM.

**LOCATION\_UPLOAD:** Stores the pending requests for information generation. The requests contain the necessary location key information, the action that resulted in the request being queued (i.e. Add, Delete, Modify), and a status field and timestamp to show progress. The requests in this table result in the location information being generated and formatted into the XML messages that are to be transmitted by the BUS.

**LOCATION\_QUEUE:** This table stores the formatted XML messages prior to their being transmitted by the BUS. Each record represents a self-contained message with data pertaining to a single location.

## Space Location messages

There are three messages that pertain to Space Locations publishing:

Message Short Name...	Belonging to Message Family Name
SpaceLocsCre	SpaceLocs
SpaceLocsMod	SpaceLocs
SpaceLocsDel	SpaceLocs

## Message family managers

This section describes the message family manager(s) (MFM) and the XML Builder Procedure for Space Locations.

**RDMFM\_SPACELOCS**– This MFM retrieves XML messages from the message queue. It contains the public procedure GETNXT, which calls SPACE\_LOCATION\_BUILD\_XML to process pending upload requests and then gets the next message waiting on the queue.

**SPACE\_LOCATION\_BUILD\_XML** – Reads the Upload Table and populates the message queue table with the generated XML message data.

## Message summary

The following table lists each message, by descriptive name and short name, along with the document type definition (DTD) used for validation during the message creation process, and the mapping document that describes the data contained in the message.

Message Description	Message Short Name	Type (DTD)	Mapping Document
Space Locations Create	SpaceLocsCre	SpaceLocsDesc.dtd	Map_Location.xls
Space Locations Modify	SpaceLocsMod	SpaceLocsDesc.dtd	Map_Location.xls
Space Locations Delete	SpaceLocsDel	SpaceLocsRef.dtd	Map_Location_Ref.xls

## SKU OPTIMIZATION subscription

SKU Optimization Information is used to receive recommended slotting information from a third-party item optimization vendor.

This family of messages is considered to be Foundation Data. Foundation Data indicates that the data is used as the basis for building other data models and is routed to every RDM installation in the enterprise.

## SKU OPTIMIZATION message structure

The SKUOptm family of messages consists of one message type: create. This message is used to populate the RDM TASK\_QUEUE table. This message is in a single node structure

## Message subscription process

The following is a description of the SKU OPTIMIZATION message subscription process:

- 4 The RDM SKU Optimization adapter recognizes that a message with the SKU Optimization-specific name (i.e. SKUOPTMCRE) exists on the RIB.
- 5 The adapter calls the public PL/SQL procedure to “consume” the message. The public “consume” procedure is named:
  - RDMSUB\_SKU\_OPTMCRE.CONSUME
- 6 The public procedure calls a set of generic consuming procedures and packages. There are four generic functions:
  - MESSAGE\_OBJECT
  - MESSAGEPRETABLE
  - MESSAGEPOSTTABLE
  - MESSAGEPOSTMESSAGE

These procedures and packages perform all of the parsing, additional processing and validation against the database.

The list of procedures and packages described in this overview is not exhaustive. For a more detailed view of them, see the RDM Generic Subscription Messages in this guide.

## Message summary

All SKU Optimization messages belong to the SKUOptm message family. The following table lists the single message that RDM subscribes to by its short name along with the document type definition (DTD) used for validation and parsing of the data during the message subscription process and the mapping documents that describes the data contained in the message.

Message Name(API)	Type (DTD)	Mapping Document
SKUOPTMCRE	SKUOPTMDesc.dtd	Map_SKUOPTMDesc.xls

Check the Retek 10 Integration Guide to view the DTD and mapping document that pertains to the message that interests you.

## PL/SQL procedures

This section describes the procedures and functions listed in the earlier “Message subscription process” section.

### Public procedures

The RIB calls the public consume procedure that, in turn, passes the message to RDM’s SKU OPTMIZATION adapter:

**RDMSUB\_SKUOPTMCRE.CONSUME**-This procedure accepts an XML file in the form of an Oracle CLOB data type from the RIB. This message contains a Sku Optimization create message.

## RDM generic processing

The list of procedures and packages used for generic processing perform all of the parsing, additional processing, and validation against the database. For a more detailed view of the generic procedures and packages, see the RDM Generic Subscription Messages in this guide.

## Primary SKU OPTMIZATION tables

The following descriptions are for the primary tables in RDM that hold Sku Optimization data:

- TASK\_QUEUE

Detailed descriptions of these tables are in the RDM Data Model document.

## Items

RDM is responsible for communicating items that reside in a forward picking / forward case picking location to a third-party SKU profiling system for the purposes of warehouse optimization.

RDM item information can be published in one of two ways. The first is through a Distribution Center (DC) Profiling support function provided within RDM. The second is through a series of item related event triggers that result in the item data being sent. These event triggers include:

- Modification or Deletion of an optimized and published item/SKU (item has sku\_optimized and sku\_opt\_published flags set to 'Y').
- Modification or Deletion of an optimized and published SKU's supplier information.
- Modification or Deletion of an optimized and published SKU's supplier country information.
- Modification or Deletion of an optimized and published SKU's supplier country DIM information.
- Creation, Modification, or Deletion of an optimized and published SKU's association to a forward picking/case picking location.
- First association of an optimized SKU to an appointment.
- First association of an optimized SKU to a allocation.

The information sent to the third-party system for add or modify requests includes: item header information, item supplier information, item supplier country information, and item supplier country DIM information. Deletes of item information will include only the DC Destination ID (from location) and the item id for the item being deleted.

## Item tables

The following tables are populated when the item information is published by RDM.

**ITEM\_MASTER\_UPLOAD:** Stores the pending requests for information generation. The requests contain the necessary item key information, the action that resulted in the request being queued (i.e. Add, Delete, Modify), and a status field and timestamp to show progress. The requests in this table result in the item information being generated and formatted into the XML messages that are to be transmitted by the BUS.

**ITEM\_MASTER\_QUEUE:** This table stores the formatted XML messages prior to their being transmitted by the BUS. Each record represents a self-contained message with data pertaining to a single item.

## Item messages

There are two messages that pertain to item publishing:

Message Short Name...	Belonging to Message Family Name
ItemWHCre	ItemWH
ItemWHDel	ItemWH

## Message family managers

This section describes the message family manager(s) (MFM) and the XML Builder Procedure for Items.

**RDMMFM\_ITEMWH**– This MFM retrieves XML messages from the message queue. It contains the public procedure GETNXT, which calls ITEM\_BUILD\_XML to process pending upload requests and then gets the next message waiting on the queue.

**ITEM\_BUILD\_XML** – Reads the Upload Table and populates the message queue table with the generated XML message data.

## Message summary

The following table lists each message, by descriptive name and short name, along with the document type definition (DTD) used for validation during the message creation process, and the mapping document that describes the data contained in the message.

Message Description	Message Short Name	Type (DTD)	Mapping Document
Items from the warehouse create	ItemWHCre	ItemWHDesc.dtd	Map_ItemMstr.xls
Item Master Delete	ItemWHDel	ItemWHRef.dtd	Map_ItemMstrDel.xls





## Chapter 4 - Subsystem interfaces

### Batch file formats

All batch files passed between an outside system and Retek Distribution Management consist of one or more records in the upload or download files. These records contain printable ASCII characters (with space characters between each field) and are of a fixed length based on the transaction type.

Fields that are defined within transaction records have an associated template that defines the arrangement, length, and logical content of the field. They appear as one of the following types:

Template	Meaning
A	A character data type.
N	A numbered digit (0 through 9).
N(p)	An unsigned p-digit number.
N(p)vN(q)	A fixed point number with a decimal point, p digits to the left of the decimal and q digits to the right.
sN(p)	A p-digit number that has a sign ('+' or '-') as its first significant character.
X	An alphanumeric character.
X(p)	A p-character string.
YYYYMMDDHHMM	A date/time, with a 4-digit year followed by a 2-digit month followed by a 2-digit day followed by a 2-digit hour, a 24 hour format, followed by a 2-digit minute.

**Note:** Numeric fields are always right justified with leading zeros. Character fields are left justified with trailing blanks, unless otherwise stated.

## Unit pick system files

### Allocation data download

This file specifies the outstanding store orders to be fulfilled to the Unit Pick System.

Field Description	Template	Destination
facility id (dc)	X (2)	Code for the DC
unit pick system code	X (4)	Code for Unit Pick System
wave number	N (3)	Unique identifier of wave
item id	X (25)	Unique identifier of the item.
dest id	X (10)	Identifier of the ship destination.
unit qty	N (8) v N (4)	Number of units
logical chute	X (10)	Logical chute assigned to group
group id	N (4)	Identifier for a set of orders
slot	N (3)	Identifier with a group associated to an order
to container id	X (20)	System generated container ID merchandise is to be packed into

### Inbound carton download

This file specifies the carton content and the associated wave to the Unit Pick System.

Field Description	Template	Destination
facility id (dc)	X (2)	Code for the DC
unit pick system code	X (4)	Code for Unit Pick System
wave number	N (3)	Unique identifier of wave
container id	X (20)	Unique identifier of the source container.
item id	X (25)	Unique identifier of the item.
requested unit qty	N (8) v N (4)	Number of units

## Process UPS upload

This file serves as a notification from a Unit Pick System to RDM concerning contents of a picked container, the associated wave number and the outbound destination ID.

Field Description	Template	Destination
Facility ID (DC)	X (2)	Code for the DC
Transaction Date/Time	YYYYMMDD HH24MI	Date and time this record was created.
Wave Number	N (3)	Unique identifier of wave
Container ID	X (20)	Unique identifier of the container.
Item ID	X (25)	Unique identifier of the item
Distributed Unit Qty	N (8) v N (4)	Number of distributed units
Dest ID	X (10)	Identifier of the ship destination.

## Pick By Light interface

The Pick By Light system (PBL) requires a variety of information from a host in order to drive its paperless picking processes. These transactions are sent periodically; the frequency is determined by the urgency of the transaction type. The host can be either Retek Distribution Management or, as in standalone operations, some other application. Data is exchanged through text files. With text file data exchange, PBL does not need to be concerned with the specifics of how the files were created or how they arrived in the upload or download directories. Each customer selects an approach to suit the preferred communication methods.

## Files and directories

All download files are placed in a directory that is named by the UNIX environment variable "DOWNLOAD\_DIR". All upload files are placed in a directory that is named by the UNIX environment variable "UPLOAD\_DIR".

The download and upload files have set names as listed in the following table. They are listed in the order in which they should be run because each download may depend upon a previous one.

Interface Name	Script Name	File Name
Destination Container Download	dest_container_download.sh	dest_container_download.dat dest_cont_item_download.dat
Distribution Item Download	distro_item_download.sh	distro_item_download.dat
Inventory Adjustment Download	inv_adj_download.sh	inv_adj_download.dat
Ship Destination Download	pps_ship_dest_upload.sh	ship_dest_upload.dat
Distro Item Upload	create_distro_item_upload.sh	distro_item_upload.dat
Expected Source Container Upload	create_exp_container_upload.sh	exp_container_upload.dat
Source Container Upload	generate_source_container_upld.sh	source_container_upload.dat

## Download transactions

The PBL downloads include several fields that are future use. These fields are included to allow for the future growth in Retek Distribution Management and to allow the PBL to work standing alone, without Retek Distribution Management. PBL download errors are recorded in the local Retek Distribution Management error log, and are not uploaded to the host. You can view and maintain this log in the Error Log screen.

## Destination container download

The Destination Container download files are built by PBL for use by Retek Distribution Management. They contain PBL built containers and the items and quantities in them that are going to be added back to inventory or shipped. If the destination is marked as the DC, the container is sent to stock; otherwise, a distribution is assumed and the container is routed appropriately. When PBL has finished creating the files, they are first copied to the download directory by PBL. Then, the script in Retek Distribution Management for this download is started by PBL. The script reads the files, loads the data into Retek Distribution Management, and adds the container information to Retek Distribution Management.

The Destination Container Download consists of a Header file and a Detail file.

The Header file, which describes the container, has the following format:

Field Description	Template	Description
Transaction Date/Time	YYYYMMDD HH24MISS	Date and time this record was created (future use).
Record Type	A	Record type in PBL. Always a 'Z' for a Destination Container Download header record (future use).
Facility ID	X (2)	Identifier for the facility.
Company Number	N (1)	Company Number (future use).
Destination ID	X (10)	Identification of the ship destination.
Container ID	X (20)	Identifier for the container.
Destination Name	X (30)	Descriptive name of the ship destination (future use).
Address 1	X (30)	First address line of the ship destination (future use).
Address 2	X (30)	Second address line of ship destination (future use).
Address 3	X (30)	Third address line of the ship destination (future use).

The Detail file, which describes the contents of the closed picking container, has the following format:

Field Description	Template	Description
Transaction Date/Time	YYYYMMDD HH24MISS	Date and time this record was created (future_use).
Record Type	A	Record type in PBL. Always a 'Y' for a Destination Container Download detail (future use).
Facility ID	X (2)	Identifier for the facility.
Container ID	X (20)	Identifier for the container.
Distribution/Order Number	X (10)	Identifier for the distribution or order.
Item ID	X (25)	Identifier for the item.
Unit Qty	N (8) v N (4)	Unit quantity that was picked for this item.
Item Description	X (60)	Text description of the item (future use).

Errors due to data integrity with the download are recorded in the error log and the record is ignored. Possible errors include:

- Facility ID does not exist in Retek Distribution Management.
- Container already exists in Retek Distribution Management.
- Non-existent Destination ID.
- Duplicate Item ID/Distro Nbr (or Item/Order) on the container detail.
- Non-existent Item ID.

## Distribution item download

The Distribution Item Download file is built by PBL and sent to Retek Distribution Management. Therefore, pick directive records can be deleted and stock allocations adjusted as needed. When PBL has finished creating the files, they are first copied to the download directory by PBL. Then, the script in Retek Distribution Management for this download is started by PBL. The script reads the file, loads the data into Retek Distribution Management, and updates the picking information in Retek Distribution Management as required.

The format for the Distribution Item Download is as follows:

Field Description	Template	Description
Transaction Date/Time	YYYYMMDD HHMISS	Date and time this record was created (future use).
Record Type	A	Record type in PPS. Always a 'X' for a Destination Container Download header record (future use).
Facility ID	X (2)	Identifier for the facility.
Company Number	N (1)	Company Number (future use).
Distro Number	X (10)	Identifier for the distribution or order.
Item ID	X (25)	Identifier for the item.
Destination ID	X (10)	Identifier for the shipping destination.
Requested Unit Qty	N (8) v N (4)	Number of units of this item requested for picking.
Distributed Unit Qty	N (8) v N (4)	Number of units of this item actually picked.

Errors due to data integrity with the download are recorded in the error log and the record is ignored. Possible errors include:

- Non-existent Facility ID.
- Non-existent Destination ID.
- Non-existent Item ID.
- No pick for the distro/item/destination.

## Inventory adjustment download

The Inventory Adjustment Download file is built by PBL and sent to Retek Distribution Management when there is a difference between the quantity sent on the Source Container Upload and the actual quantity picked. RDM validates the data in the file and sends the information in an Inventory Adjustment Upload to the host system. This is the only action RDM takes on this; no change in RDM data occurs. After PBL creates the files, they are copied to the download directory by PBL. Then, PBL starts the script in Retek Distribution Management, or this download. The script reads the files, validates the data, and inserts the information into the Inventory Adjustment Upload table in RDM for upload to the host (reason code to the host for this adjustment is 30).

The format for the Inventory Adjustment Download is as follows:

Field Description	Template	Description
Transaction Date/Time	YYYYMMDD HHMISS	Date and time this record was created (future use).
Record Type	A	The record type in PBL. This is always sent as 'W' for an Inventory Adjustment Download (future use).
Facility ID	X (2)	Identifier for the facility.
Company Number	N (1)	A single digit number for the company. This is a new system parameter in Retek Distribution Management (future use).
Distro Number	X (10)	The identifier for the distribution (future use).
Item ID	X (25)	The identifier for the item.
Adjusted Unit Qty	sN (8) v N (4)	The difference between source container units and the number of units of this item actually picked. A positive number means more were picked than expected; a negative number means fewer were picked than expected.

Errors due to data integrity with the download are recorded in the error log and the record is ignored. Possible errors include:

- Non-existent Item ID.



## Upload transactions

### Ship destination upload

The Ship Destination Upload file is spooled from the Shipping Destination table and sorted by Facility ID, Company Number, and Shipping Destination. This file is empty unless an adjustment action (add/modify/delete) is sent to Retek Distribution Management from the host, or performed in the Destination Editor screen. Whenever an adjustment is performed, all shipping destinations that Retek Distribution Management knows about are sent to PBL via the upload. Thus, this upload is an all or nothing data file.

The format for the Ship Destination Upload is as follows:

Field Description	Template	Description
Transaction Date/Time	YYYYMMDD HHMISS	Date and time this record was created.
Record Type	A	Record type in PBL. Always a 'A' for a Ship Destination Upload.
Facility ID	X (2)	Identifier for the facility.
Company Number	N (1)	Company Number.
Destination ID	X (10)	Identification of the ship destination.
Destination Name	X (30)	Descriptive name of the ship destination.
Address 1	X (30)	First address line of the ship destination.
Address 2	X (30)	Second address line of ship destination.
Address 3	X (30)	Third address line of the ship destination.

## Distro item upload

The Distro Item Upload file contains records that indicate to PBL which items, and how many, should be shipped to specified destinations. After the distribution process runs, this file is built from all remaining sorted allocation records that are eligible to be processed by PBL (the item does not have a forward picking location defined). Records are sorted by facility number, company number, distribution/order number, item, distro/order creation time stamp, and shipping destination.

The format for the Distro Item Upload is as follows:

Field Description	Template	Description
Transaction Date/Time	YYYYMMDD HHMISS	Date and time this record was created.
Record Type	A	Record type in PBL. Always a 'B' for a Distro Item Upload.
Facility ID	X (2)	Identifier for the facility.
Company Number	N (1)	Company Number.
Distro Number	X (10)	Identifier for the distribution or order.
Item ID	X (25)	Identifier for the item.
Distro Create Date/Time	YYYYMMDD HHMISS	Date and time the distro/order was created.
Destination ID	X (10)	Identifier for the shipping destination.
Unit Qty	N (8) v N (4)	Number of units of this item to be shipped.
Item Dept	X (4)	Department of the item.
Item Description	X (60)	Item Description.

## Expected source container upload

This Expected Source Container Upload file contains records identifying all Inventory containers necessary to fulfill the PBL requirements determined by the last distribution run. This information is used by PBL to "know" ahead of time what containers will be needed by PBL. This file is built after each distribution run. Records are sorted by facility number, company number, distribution/order number, item, distro/order creation time stamp, and container ID.

The format of the Expected Source Container Upload is as follows:

Field Description	Template	Description
Transaction Date/Time	YYYYMMDD HH24MISS	Date and time this record was created.
Record Type	A	Record type in PBL. Always a 'C' for an Expected Source Container Upload.
Facility ID	X (2)	Identifier for the facility.
Company Number	N (1)	Company Number.
Distro Number	X (10)	Identifier for the distribution or order.
Item ID	X (25)	Identifier for the item.
Distro Create Date/Time	YYYYMMDD HH24MISS	Date and time the distro/order was created.
Container ID	X (20)	Identifier for the container.
Requested Qty	N (8) v N (4)	Unit quantity that will be requested for picking of this item.

## Source container upload

The Source Container Upload file is built as PBL Source containers are picked and dropped off at the PBL staging area. The upload file is used to match up expected containers with actual source containers delivered to PBL. It has no sorted order.

**Note:** The value of Actual Quantity in the Upload will be 0 (zero) if the pick was 'canceled' either by the user or indirectly via a system function (such as a location marked for cycle count).

The format for the Source Container Upload is as follows:

Field Description	Template	Description
Transaction Date/Time	YYYYMMDD HH24MISS	Date and time this record was created.
Record Type	A	Record type in PBL. Always a 'D' for a Source Container Upload.
Facility ID	X (2)	Identifier for the facility.
Container ID	X (20)	Identifier for the container.
Actual Qty	N (8)	Unit quantity in the container.

## Sortation subsystem interface

Due to the increased use of UCC-128 labeled containers and the addition of WIP code functionality to Retek Distribution Management, the Retek Distribution Management Sortation module will now send "container divert instruction" messages to the sortation system to control the flow of containers on the conveyor.

Each message that Retek Distribution Management sends to the sortation system informs it of the next logical destination for a container. The divert instruction could be, but is not limited to, one of the following: any type of processing area, QA sampling, palletization, putaway staging, or shipping lane divert instructions. Initially, a message is sent to the sorter whenever a container is created on Retek Distribution Management. However, subsequent messages can be sent to the sorter if the container is assigned one or more WIP codes. The sortation system is only sent the next logical destination for a container.

The sortation system continues to notify Retek Distribution Management of any container diverts that occur on the conveyor system. Depending on the type of divert that has taken place, Retek Distribution Management either attempts to auto-receive, move, or manifest the container.

## Files and directories

All files are placed in a directory that is named by the UNIX environment variable 'SORTATION\_DIR'. The files have set names as listed below. They are listed in the order in which they should be run because each download may depend upon a previous one.

Interface Name	Script Name	File Name
Container Divert Download	sorter_downld.sh	sorter_downld.dat
Container Divert Instruction Upload	sorter_upload.sh	sorter_upload.dat

## Download transactions

### Container divert message

The sortation system sends a message when a container is scanned, indicating whether it was scanned as an induction, diverted to a processing area, or diverted to a shipping lane. If the container was inducted, Retek Distribution Management performs an auto-receiving function. If the container is diverted to a processing area, Retek Distribution Management updates the location of the container. When a divert to a shipping lane is sent, Retek Distribution Management adds the container to the manifest for the trailer if one is available. For more details, refer to the Retek Distribution Management Shipping Module in the *Retek Distribution Management User Guide*.

The Container Divert Message Download file has the following format:

Field Description	Template	Description
Container ID	X (20)	Identifier of the container.
Divert Type	A	I = Induction D = Shipping Lane Divert.
Logical Destination	X (4)	Area of the DC to which the container was sorted.
Tracking ID	X (25)	Tracking ID (if any) applied to the container by a carrier for tracking purposes.
Divert Timestamp	YYYYMMDD HH24MISS	Date/Time the container was scanned by the sortation system.

## Upload transactions

### Container divert instruction message

Retek Distribution Management sends a Container Divert Instruction Message to the sortation system to control the flow of containers on the conveyor system. If a container must be diverted to several areas in the distribution center before it is ready to be putaway or shipped, Retek Distribution Management will only inform the sortation system of the next logical destination for the container. This way, the sortation system does not need to keep track of all divert instructions for a container. The first divert instruction for a container is sent when the container ID is first created on Retek Distribution Management.

When the receiving allocation module creates a container, Retek Distribution Management calculates a pallet group identifier in order to give a palletization operator a quickly recognized code that helps to group cartons together on pallets. Retek Distribution Management assigns a four-digit number to each PO/Item/Destination and concatenates this with the total number of cartons expected for the pallet group to make up the pallet group identifier.

The Container Divert Instruction Message Upload file has the following format:

Field Description	Template	Description
Container ID	X (20)	Container identifier.
Logical Destination	X (4)	Next area of DC to which the container should be diverted.
Transaction Date/Time	YYYYMMDD HHMISS	Date/Time of upload to sortation system.

## Manifest mailing system

The Manifest Mailing System uses the merchandise carton ID to form an Oracle Data Base Compliant (ODBC) query into the Retek Distribution Management data base. This query gathers the information necessary for generating a shipping label and manifesting the carton.

Merchandise, planned and picked, using the logic currently implemented in Retek Distribution Management, is first taken to a shipping station. Each shipping station is a PC running a manifest mailing system with interfaces to a user interface terminal and often to a scale.

The label applied by the Retek Distribution Management picker is then scanned to retrieve the carton ID necessary for the ODBC query.

## Files and directories

In addition to the normal BOL upload records, MMS information is uploaded to the host. This additional information is prepared for upload to the host system upon completion of the normal BOL upload operation.

Each Container ID, pro number combination in the shipment has one detail record in the MMS upload. The BOL Sequence Number is incremental and unique for each BOL.

An MMS upload consists of a single detail file with the following format:

Field Description	Template	Description
Batch Number	N (7)	Numeric Sequence of the upload
BOL Number	X (17)	BOL number
Container ID	X (20)	Identifier of container
Pro Number	X (18)	Shipper's tracking number
Cube	N (10) v N (2)	Container cube
Weight	N (8) v N (4)	Container weight
Freight	N (6) v N (2)	Freight charge
Markup	N (6) v N (2)	Markup charge
Charge type	X (6)	Carrier charge code
Service Code	X (6)	Carrier service code
Service Level	X (12)	Carrier service level
Tracking ID	X (25)	Tracking ID

## Views

The Retek Distribution Management data base, which the Manifest Mailing System queries is actually two views: the MMS Container View and the MMS Container Item View.



## MMS container view

The value in the "CARRIER\_SERVICE\_CODE" is set by the host system, or it could be blank. If the value is blank, the user must input data at the shipping station. The Manifest Mailing System can change the value of the CARRIER\_SERVICE\_CODE, even if the field is not blank.

Retek Distribution Management downloads the SHIP\_TO\_ADDRESS with the stock order. If this field is blank in the order download, the information will be supplied by the SHIP\_DEST table. The Manifest Mailing System can change the value of the SHIP\_TO\_ADDRESS, even if the field is not blank.

The format of the MMS Container view is as follows:

Field Description	Template	Description
Facility ID	X (2)	Identifier for the facility
Container ID	X (20)	Identifier for the container
Ship Address Description	X (30)	The description (such as store or ship-to name). This is the first line of the address block.
Ship Address1	X (30)	Shipping Address Line 1
Ship Address2	X (30)	Shipping Address Line 2
City	X (25)	Shipping City
State	X (3)	Shipping State
Zip	X (10)	Shipping Zip
Dest ID	X (10)	Destination identifier
Carrier Service Code	X (6)	Carrier service code for the delivery (such as First Class)
Bill Address Description	X (30)	The first line of the address block. A description, such as company or bill-to name.
Bill Address1	X (30)	Billing Address line 1
Bill Address2	X (30)	Billing Address line 2
Bill Address3	X (30)	Billing Address line 3
Amount1	N (8) v N (4)	Amount Charge 1
Amount2	N (8) v N (4)	Amount Charge 2
Amount3	N (8) v N (4)	Amount Charge 3
DL Comment	X (30)	Download comment that will be printed on the label (optional)

**Note:** A default value using 'nvl' or decode statements should be supplied for any null values.

## MMS container item view

The data item called "DISTRO" is analogous to "CUSTOMER\_ORDER\_NUMBER" in a wholesale system.

The format of the Container Item View is as follows:

Field Description	Template	Description
Facility ID	X (2)	Identifier for the facility
Container ID	X (20)	Container identifier
Item ID	X (25)	Unique item identifier
Unit Qty	N (8) v N (4)	Standard unit quantity for an item
Weight	N (8) v N (4)	Item weight or unit quantity weight
Retail Price	N (16) v N (4)	Retail selling price
Class	X (7)	Class of merchandise (optional)
Distro/Order	X (10)	Unique identifier of a distribution or order
Ticket Type	X (4)	Refers to Ticket Type table (optional)

## Tables

The Manifest Mailing System populates two tables: the MMS Manifest Table and the MMS Container Table.

### MMS manifest table

Retek Distribution Management and MMS work congruently to generate each manifest. Before a new grouping of containers is started, PRO\_NBR (pickup) numbers are assigned by MMS. Next, a row is inserted into the MMS MANIFEST table and the STATUS is set to 'OPEN'. When MMS inserts a row into the MMS MANIFEST table, Retek Distribution Management looks up the corresponding BOL number in the RETEK DISTRIBUTION MANAGEMENT MANIFEST table. If the BOL does not exist, Retek Distribution Management inserts a MANIFEST record with a new system-generated BOL number (PRO\_NBR=PICK\_UP\_NBR, DEST\_ID = (SCP Mixed\_dest\_id)).

When a shipment is released by MMS and/or MMS is ready to have a BOL uploaded, the 'STATUS' in the MMS\_MANIFEST table is updated to 'SHIPPED'. Retek Distribution Management recognizes the status change and administers normal ship operations (adjust manifest, set container\_status = 'S' for all containers for BOL (pick\_up\_nbr), upload BOL).

The MMS\_MANIFEST table format is as follows:

Field Description	Template	Description
Facility ID	X (2)	Identifier for the facility
PRO NBR	X (18)	Pickup number
MANIFEST STATUS	X (10)	Status of manifest: <ul style="list-style-type: none"> <li>• OPEN: a new grouping of containers is started</li> <li>• SHIPPED: shipment is released</li> </ul>
Trailer ID	X (12)	Identifier for trailer
Carrier Code	X (4)	Code of the carrier for the order

**Note:** If TRAILER\_ID does not exist, create TRAILER and CARRIER records.

Set:

TRAILER\_STATUS = 'LOADING'

TRAILER\_CUBE=SCP

'default\_trailer\_cube'=CARRIER\_CODE

CARRIER\_NAME=CARRIER\_CODE

## MMS container table

Operators scan the containers and the Manifest Mailing System writes records to the MMS\_CONTAINER table. Retek Distribution Management uses this information to set the CONTAINER\_STATUS to 'M' (manifested) and update the CUBE and WEIGHT for each record inserted into the MMS\_CONTAINER table.

The MMS\_CONTAINER table is as follows:

Field Description	Template	Description
FACILITY ID	X (2)	Identifier for the container
CONTAINER ID	X (20)	Identifier for the container
PRO NBR	X (18)	Pickup number
CONTAINER CUBE	N (6) v N (2)	Container cube (dimensionless)
CONTAINER WEIGHT	N (4) v N (3)	Container weight (dimensionless)
FREIGHT CHARGE	N (6) v N (2)	Freight charge.
MARKUP CHARGE	N (6) v N (2)	Markup charge.
CHARGE TYPE	X (6)	Billing Method (such as COD and 30-day invoice).

Field Description	Template	Description
CARRIER SERVICE CODE	X (6)	Carrier service code for the delivery (such as First class).
SERVICE LEVEL	X (12)	Code for shipment (Next Day, Second Day Air).
TRACKING ID	X (25)	Tracking Identifier
CREATION TS	YYYYMMDDHH24MI	Date and Time record created

## Kewill shipping system Interface

RDM interfaces with Kewill, shipping and transportation management system. As outbound cartons are created in RDM during the distribution process, information about them will be sent to Kewill so that Kewill can prepare a shipping label and other rate information. Kewill will send RDM the shipping label, which will be stored as a database field.

Kewill provides a shipping system for managing the use of common carriers (i.e. Fed-Ex, UPS, etc.) This interface will include the TCP/IP sockets layer between RDM and the Kewill K-Ship shipping information such as name, shipping address, package dimension, and estimated weight to the Kewill K-Ship system. K-Ship will respond with the estimated rate, tracking number, and label information to be stored in RDM database. When the actual weight is determined, RDM will send a message to K-Ship, and a response is returned with the actual rate and tracking number to be loaded into the RDM database. Once the package is actually loaded to be shipped, RDM will send a message to K-Ship so that it can update its manifest information.

## Triggers

### **ship\_carton\_trg**

This new trigger will notify Kewill when the status of outbound cartons have been updated to 'S'hipped.

### **create\_sorter\_instructions\_trg**

When the Kewill interface is enabled in RDM, this trigger will call the socket interface package, on the downloads from RDM to Kewill.

## Packages

### **label\_info\_received**

This stored procedure is used to receive shipping label information from the Kewill system. This information will be inserted in database tables within RDM.

### **package\_weighed**

This stored procedure is used to receive outbound carton tracking numbers once Kewill receives that actual weight of these containers. The tracking number will be updated on the outbound carton record in RDM.

### **ship\_lane\_upload**

This stored procedure is used to call the Kewill socket interface procedure, case\_weighted, and notify Kewill of the actual container weight.

## Tables

### Cont\_Ship\_Label

RDM table to hold shipping label information and error message returned from Kewill. This will be a child table to the container table, (facility\_id and container\_id must exist in the container table).

Column Name	Column Type	Primary Key?	Req?	Valid Values	Description
facility_id	VARCHAR2(2)	Y	Y		Facility ID
container_id	VARCHAR2(20)	Y	Y		Unique container ID on shipping label.
binary_label_info	LONG	N	N		Carrier compliant shipping label for outbound carton
label_type	VARCHAR2(4)	N	N		Type of label – EX: 'PNG ', 'ZEBR', 'MONA'
label_size	NUMBER	N	N		Size in bytes of the binary information contained in binary_label_info
rejection_reason	VARCHAR2(50)	N	N		Error message returned from Kewill when Kewill is unable to create a shipping label for the container

## Rapistan Socket Interface

RDM interfaces with Rapistan through socket interfaces. RDM still generates directives based on logical dest ID's associated to locations setup in RDM.

For RDM generated directives (message sent to the control system), a trigger will call stored procedures used in the socket interface for various message types. Different message types are generated depending on where the container is going in the facility due to data required by the control system. RDM will determine the message type and call the appropriate procedure.

For control system confirmations (message received from the control system), divert confirmations will be sent to RDM via stored procedures. Similar to the directive procedures, the upload confirmation procedures will be created based on the message type sent from the control system.

### Triggers

#### **create\_sorter\_instruction\_trg**

When the Rapistan interface is enabled in RDM, this trigger will call the socket interface package, on the message transfer from RDM to Rapistan.

#### **appt\_rec\_dir\_trig**

When the Rapistan interface is enabled in RDM, this trigger will write receiving directive records for ASN appointments when the appointment status is updated to 'PEND'.

#### **cont\_dest\_trg.sql**

Send the dest ID (carrier service) to Rapistan if the dest ID changes for an outbound container.

### Packages

#### **process\_diverts\_a.sql**

Select additional fields from the sorter\_intake table, in addition to performing palletization logic.

#### **receiving\_upload.osp**

Accept receiving location directive confirmations from the control system. It will insert records into the sorter\_intake table to be processed by the process\_diverts\_a.sql script.

#### **divert\_confirmation.osp**

Accept divert directive confirmations from the control system. It will insert records into the sorter\_intake table to be processed by the process\_diverts\_a.sql script.

**ship\_lane\_upload.osp**

Accept shipping location directive confirmations from the control system. It will insert records into the sorter\_intake table to be processed by the process\_diverts\_a.sql script.

**pack\_wave\_release\_upload.osp**

Receive pack wave release confirmations from the control system. It will call the new unit\_sorter\_directive.osp stored procedure to send unit sortation information to the control system for the pack wave that was released by the control system.

**unit\_control\_sorter\_upload.osp**

Receive unit sorter confirmations from the control system. It will update the container\_item table for the outbound carton being sorted.

**combine\_wip\_codes.osp**

WIP processing associated with outbound cartons.

**receive\_container2.osp**

Write receiving\_directive records upon receipt for non-ASN, specified case pack PO receiving.

## Tables

### Sorter Intake

This table is used for all container transactions from the control system to RDM, including inbound, outbound, and movements within the facility.

Field Name	Field Type	Primary Key?	Req?	Description
facility_id	X (2)	N	N	Facility identifier
sorter_seq	N (9)	Y	Y	Sorting Sequence
container_id	X (20)	Y	Y	RDM container identifier
logical_dest_id	X (4)	N	Y	Logical destination ID that relates to a location within RDM.
divert_type	X (1)	N		
divert_ts	DATETIME	N	N	Date/timestamp
tracking_id	X (25)	N	N	Current field.
pallet_id	N (6)	N	N	Rapistan pallet identifier.
expected_container_qty	N (6)	N	N	Number of cases on pallet ID.

Field Name	Field Type	Primary Key?	Req?	Description
scale_weight	N (4) v N (3)	N	N	Scale weight
Length	N (4) v N (2)	N	N	Measured length
Width	N (4) v N (2)	N	N	Measured width
Height	N (4) v N (2)	N	N	Measured height
Packer_id	X (10)	N	N	Populated for shipping cartons for audit purposes.