Retek® Merchandising System 10.1.4

Release Notes

Retek®

WWW.RETEK.COM

HELPING THE RETAIL INDUSTRY CREATE, MANAGE AND FULFILL CONSUMER DEMAND™

Corporate Headquarters:

Retek Inc.
Retek on the Mall
950 Nicollet Mall
Minneapolis, MN 55403

888.61.RETEK (toll free US)
+1 612 587 5000

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

Printed in the United States of America.

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.

Contact Method	**Contact Information**
Internet (ROCS)	www.retek.com/support Retek's secure client Web site to update and view issues
E-mail	support@retek.com
Phone	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
Mail	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.

Overview

Note: The RMS 10.1.4 Release is Mandatory for all 10.0 customers.

Other Mandatory upgrades, at this time, apply to the following products:

- SeeBeyond ® eGate™ 4.5.3;
- Oracle® Database™ 9.2;
- Retek® Integration Bus™ (RIB) 10.3.1;
- Retek® Merchandising System™ (RMS) 10.1.4;
- Retek® Distribution Management™ (RDM) 10.3.1

When Retek Customer Support investigates a potential software bug, they assume that you have made **all** required upgrades and applied **all** patches for a given module before the issue was discovered. This Patch CD contains the 10.1.4 patch release. Before you apply the RMS 10.1.4 patch:

- Check that RMS 10.1 is installed.

Also check that all the following patches have been applied:

- RMS 10.1.2
- RMS 10.1.3

The 10.1.4 patch contains files that were modified since the previous patch release. For detailed information on what is included regarding software fixes in this patch release, refer to the 10.1.4 patch documentation located in the doc folder.

Before applying the patch source files over your code:

- Note whether customizations have been made to the module. If so, then the customizations must be reapplied to the new version of the module (or the fix may need to be applied to the custom version of the code).
- Copy the original files to a different directory before you copy over them in case you need to refer to them at a later date.

DEFECT documentation (DEFECT MODULE XREF 1014.xls)

A DEFECT fix is a modification to the base Retek code (e.g. a bug fix, a performance enhancement, or a functional enhancement). Each DEFECT fix that is included in this patch has a corresponding DEFECT document in the \doc\defect_doc folder titled <DEFECT#> <module>.doc, such as “123456 nxprcno.doc”. There is also a document in that same folder titled “DEFECT MODULE XREF 1014.xls”, that lists every DEFECT # and the accompanying modules/scripts that are included in the patch.

To provide better customer service, we have created a more comprehensive, user-friendly version of our DEFECT module cross-reference Excel document. This updated document includes a full list of all the previous patch DEFECT modules, plus new tabs showing DEFECTs related to the current patch, and current DEFECTs broken out by module types. We hope this will assist you in installing and maintaining patch-related projects. We will follow this format for all upcoming patches.

Enclosed is an updated version of the cross-reference document. Each DEFECT document should be fully reviewed before this patch is implemented. Following is a list of the more significant fixes in the 10.1.4 patch.

Mandatory enhancements for RMS APIs to the RIB

MASTER DEFECT 344193

Publication family packages (rmsmfm_*.pls) and subscription family packages (rmssub_*.pls) were changed to use Oracle Objects. Originally, these packages used CLOBs that contained XML.

The master Oracle Object used is called RIB_OBJECT. Messages passed out of publication packages and passed into subscription packages are of the type RIB_OBJECT. All objects created in the publication and subscription packages subclass the RIB_OBJECT type. Examples of using RIB_OBJECT are shown below in the New Code section. For more details on the RIB_OBJECT type and its subclasses, refer to RIB 10.3 Documentation.

Along with changes involving Oracle Objects, the overall architecture for publication family packages has changed. Details on publication enhancements can be found in the document Header-Detail Publishing Overview.doc.

For subscription family packages, the overall architecture did not change. However, the packages called by the subscription family packages were changed to include bulk SQL operations and caching of values that are frequently queried but seldom change.

Detail Defect 343390 Allocation Publishing

Allocation publication follows the Master Detail publishing changes described in the master defect document.

Detail Defect 343955 Purchase Order Publishing

Purchase order publication follows the publishing changes described in the master defect document and the header-detail publishing overview document. There were also changes made specific to Purchase Order Publishing. The order message family and the orderphys message family were combined into one family. In the old model, the orderphys messages contained physical locations and quantities at the physical level, while order messages contained virtual locations and quantities at the virtual level. In the new model, messages contain both the physical and virtual level. Since virtual information is a subset of physical information, each physical node of a message contains one or more virtual nodes. Within a detail message, the header node is labeled ‘PODesc’, the physical location node is labeled ‘PODtl’, and the virtual location node is labeled ‘POVirtualDtl’.

Detail Defect 343956 Work Order In Publishing

The changes for Work Order In (WOIN) for the most part follow the publishing changes outlined in the master defect document.

WOIN publishing was changed to publish Oracle Objects instead of CLOBs.

WOIN publishing was changed to create the messages at GETNXT time. Previously, the complete messages were placed in the queue table in the form of a CLOB by the trigger. The updated trigger places information needed to create the message on the queue in columns. The messages are actually constructed (as Oracle Objects) during the call to GETNXT.

WOIN is similar to ALLOC, TSF, and ORDER publishing in that it must have a certain status before it can be published. However, instead of relying on the work order’s status (it does not have one), it relies on the status of the order it is associated with.

WOIN messages were previously sent out flat. They are now sent out grouped together by wo_id and order_no

Detail Defect 344012 Transfer Publishing

Transfer publication follows the Master Detail publishing changes described in the master defect document.

Changes specific to transfer publishing:

The transfers family does not have a <FAMILY>_DETAILS_PUBLISHED table. Instead it has a PUBLISH_IND on TSFDETAIL. It works this way due to the fact that all information needed for routing of a transfer message is held at the header level (the to and from locations).

Detail Defect 344021 ASNOUT (BOL) Subscription

The RMSSUB_ASNOUT (rmssub_asnouts(b).pls) package was changed to use Oracle Objects instead of a Clob as its input from the RIB. For more information, see the subscriber section in the master defect document.

The BOL_SQL (bolsqls(b).pls) package was changed for performance reasons. These changes include the addition of bulk DML statements.

Detail Defect 344026 Appointment Subscription

The RMSSUB_RECEIVING (rmssub_receivings(b).pls) package was changed to use Oracle Objects instead of a Clob as its input from the RIB. For more information see the subscriber section in the master defect document.

The RMSSUB_APPOINT (rmssub_appoints(b).pls) package was changed to use Oracle Objects instead of a Clob as its input from the RIB. For more information see the subscriber section in the master defect document.

The APPOINTMENT_PROCESS_SQL (apptprocs(b).pls) package was changed for performance reasons. These changes include the addition of bulk DML statements.

The RMSSUB_RECEIPT (rmssub_receipts(b).pls) package was changed to use Oracle Objects instead of a Clob as its input from the RIB. For more information see the subscriber section in the master defect document.

The RMSSUB_OTBMOD (rmssub_otbmods(b).pls) package was added to decouple OTB processing from the receiving processing to avoid locking issues.

The ORDER_RCV_SQL (ordrcvss(b).pls) package was changed for performance reasons. These changes include the addition of bulk DML statements. OTB processing was changed to avoid locking issues.

The STOCK_ORDER_RCV_SQL (stkordrcvs(b).pls) package was changed for performance reasons. These changes include the addition of bulk DML statements.

Detail Defect 344042 Seed Data Publication

Seed Data Publication interfaces RMS’s CODE_HEAD, CODE_DETAIL and DIFF_TYPE tables to external system. It is written using the pre-stormfin architecture (i.e. Clob messages, not Oracle Object messages).

CODE_HEAD_XML (code_head_xmls (b).pls) was added to RMS base line from Williams-Sonoma.

CODE_DETAIL_XML (code_detail_xmls (b).pls) was added to RMS base line from Williams-Sonoma.

DIFF_TYPE_XML (diff_type_xmls (b).pls) was added to RMS base line from Williams-Sonoma.

RMSMFM_SEEDDATA (rmsmfm_seeddatas(b).pls) was added to RMS base line from Williams-Sonoma.

EC_TABLE_CODEHD_AIUDR (ec_table_codehd_aiudr.trg) was added to RMS base line from Williams-Sonoma.

EC_TABLE_CODEDTL_AIUDR (ec_table_codedtl_aiudr.trg) was added to RMS base line from Williams-Sonoma.

EC_TABLE_DIFF_TYPE_AIUDR (ec_table_diff_type_aiudr.trg) was added to RMS base line from Williams-Sonoma.

Detail Defect 344141 Inventory Adjustment Subscribing

The RMSSUB_INVADJ (rmssub_invadjs(b).pls) package was changed to use Oracle Objects instead of a Clob as its input from the RIB. For more information see the subscriber section in the master defect document.

Detail Defect 344166 RMS performance – Miscellaneous

See detail document

Detail Defect 344369 Stock Order Status Subscription

The RMSSUB_SOSTATUSCRE (mssub_sostatuscres(b).pls) package was replaced with RMSSUB_SOSTATUS (rmssub_sostatuss(b).pls) package. It was changed to use Oracle Objects instead of a Clob as its input from the RIB. It was also changed for performance reasons. These changes include the addition of bulk DML statements. For more information, see the subscriber section in the master defect document.

Detail Defect 344376 ASN IN Subscription

The RMSSUB_ASNIN (rmssub_asnins(b).pls) package replaced all of the removed objects. The logic was changed to use Oracle Objects instead of a Clob as its input from the RIB. It was also changed for performance reasons. For more information see the subscriber section in the master defect document.

The ASN_SQL (asns(b).pls) package was changed for performance reasons. These changes include the addition of bulk DML statements. For more information, see the subscriber section in the master defect document.

Detail Defect 344403 Item Publishing

Add threading to item publication.

Add THREAD_NO field to ITEM_MFQUEUE.

Change RMSMFM_ITEMS.ADDTOQ to populate the THREAD_NO field. This uses the new ITEM_HASH procedure to calculate the value to use for the THREAD_NO field.

Change RMSMFM_ITEMS.GET_NXT to only publish items assigned to the passed in thread value.

Detail Defect 344624 Direct Store Delivery Consuming

DSD functionality has been pulled over from the batch stack into the RIB since RSS needs to communicate DSD to RMS and RSS publishes this info to the RIB. This RMS consumer eWay now replaces the old RMS batch module.

In the current implementation, the DSD consume PL/SQL package publishes a DSDDeals message back to the calling eWay (much like Receiving publishes an OTB message), which message is then fed through a queue to the DSDDeals PL/SQL consume package.

The reason for having two packages doing DSD consume is technical: XA sessions to Oracle are not compatible with the Oracle PL/SQL code if that code makes an extproc call. (In DSDs POs are created, and deals may be applied to these POs. The deals application code only exists in a shared batch library which Oracle calls as an extproc. Re-writing deals in PL/SQL to avoid the extproc was not feasible due to performance, dev time and functionality concerns.)

The work-around was to have the DSD consume package publish a deals message for the POs that it created and this deals message is then fed into a separate eWay (DSDDeals) which connects to Oracle in a non-XA session and essentially runs a PL/SQL command that applies deals to the orders in the message. This works, unlike making this call in an XA session.

This approach was tested for both functionality and failure recovery. It was proven to function as expected, without functional gaps and windows for unrecoverable errors. Worst case scenario could be the eWay for DSD running fine and the DSDDeals eWay failing, which would cause the DSDDeals queue to grow, but the messages would be in the error hospital just like in a failed XA eWay failure at which point admin is notified and can take the appropriate action just like it would with any other XA eWay (e.g.: OTB). Please read the related documents for full details.

Additional Base RMS functionality fixes

Bracket Costing

Defect 342610 describes that users are unable to approve multiple cost changes created when the bracket cost structure of a supplier is modified. Please read the related documents for full details.

Clearance

Defect 340301 describes that an error message is displayed when a user is creating a clearance for an item that exists in a Corporate Zone Group. The error is received when entering the form pcclsup. Please read the related documents for full details.

Complex Deals

Defect 342622 describes that users are unable to create buy/get deals where the buy or get item is on an existing deal from a different supplier. Please read the related documents for full details.

Invoice Matching

Defect 344501 describes that Invoice details are deleted upon summary matching. Please read the related documents for full details.

Defect 344305 describes that when difference between the received and invoice cost is within the tolerance, the invcpost batch program fails with the following error message: “invcpost_1~20030313102813~write_invc_tran_data~tran_data~105~RET-0105: generic stored procedure error~Error ORA-02291: integrity constraint (RMS.TDA_TDC_FK) violated - parent key not found returned by program unit STKLEDGR_SQL.TRAN_DATA_INSERT.” Please read the related documents for full details.

Defect 342720 describes that when the user goes to the Invoice VAT Details window, views a different currency (Primary or Supplier) and returns to the main window many errors are displayed. Please read the related documents for full details.

Defect 342882 describes that an invoice associated with an order that has deals applied is posted by running invcpost.pc. When the invmatch form is viewed, the Rcpt Cost and Rcpt Qty fields are equal to 0. Prior to posting these fields contained actual values. Please read the related documents for full details.

Defect 344067 describes that the Reference dropdown list box is disabled when selecting ‘Vendor Markdown’ and ‘Stand Alone Debit/Credit’. Please read the related documents for full details.

Defect 342679 describes that when entering the invoice form in NEW mode. Begin entering invoice information. Then, the user enters multiple VAT codes in NEW mode and then returns to the invoice head form. If the user clicks Receipts the error message “Please enter a VAT code.” is displayed. The field B_invc_head.TI_invc_vat_code should be disabled.

The other problem occurs when the user is in the VAT Details window, partially enters data, and then clicks back to the Invoice window without clicking Cancel or OK. This causes multiple errors. Please read the related documents for full details.

Item List

Defect 344454 describes that Item Lists cannot be created with the online forms. Please read the related documents for full details.

Defect 342398 describes that after the user clicks OK+Repeat when viewing an Item List and then selecting another Item list to view, the Total Items count on the Item List header window is not displayed. Please read the related documents for full details.

Location List

Defect 342889 describes that lclrbld.pc fails with a fatal error when location criteria doesn’t exist. Please read the related documents for full details.

Defect 342276 describes that the Location List Type dropdown list values are incorrect. Currently they display ‘Yes’ and ‘No’. The values should be ‘Static’ and ‘Dynamic’. Please read the related documents for full details.

Ordering

Defect 343107 describes that the user is receiving an error message when there are multiple origin country items on the order and the pack size of the items are greater than or equal to 1.

The form is passing over the wrong item to the called form. Please read the related documents for full details.

Defect 343574 describes that the Savepoint and Rollback commands in this form are incorrect; they include spaces in the savepoint/rollback name. Please read the related documents for full details.

Defect 343983 describes that an error occurs when choosing ‘Pallet’ as a unit of purchase from the LOV in ordmtxws.fmb. Please read the related documents for full details.

Defect 342110 describes that item LOV brings back zero records when no records exist on the DIFF_GROUP_HEAD table. Please read the related documents for full details.

Point of Sale

Defect 343372 describes that the posupld batch program fails with a deadlock error when it runs on a multi threading environment and the invoice matching indicator is turned on. Please read the related documents for full details.

Defect 342471 describes that the Credit Card Validation block stays enabled when the user Enters/Maintains a Credit Card Tender Type Group record and then edits a Tender Type Group that is not a Credit Card. Please read the related documents for full details.

Price Management

Defect 344098 describes that an ORACLE unable to UPDATE error is displayed to the user after clicking Apply when the cost change description is null. Please read the related documents for full details.

Defect 343978 describes that pcext is writing incorrect records to tran_data for price changes. No records are written to reflect the price change for the warehouse and extra records are written for some stores. Please read the related documents for full details.

Defect 343869 describes that an ORACLE unable to INSERT error is displayed to the user on submitting the price change when the required fields are null. Please read the related documents for full details.

Promotions

Defect 343970 describes that PRMEXT uses a text comparison to populate promotion code variables. Please read the related documents for full details.

RMS interface via RETL

Defect 341526 describes that in the rmsl_forecast script, the location of the input file is incorrect and the validation for the input file location does not exist. Please read the related documents for full details.

Replenishment

Defect 344289 describes that REPLADJ program calculates incorrect max stock while using floating point method. Please read the related documents for full details.

REPLENISHMENT P2

Defect 344716 describes that rplext is generating duplicate orders for the warehouse and store replenishment. Please read the related documents for full details.

Return To Vendor

Defect 343086 describes that the RTV_SQL package calls Invoice Matching packages even when Invoice Matching is turned off. Please read the related documents for full details.

Defect 343097 describes that there are inconsistencies with the RTV_ORDER_NO field lengths on the database. RTV_HEAD.RTV_ORDER_NO is defined as a NUMBER(10), INVC_HEAD.REF_RTV_ORDER_NO is defined as a NUMBER(6). The forms and database need to consistently use NUMBER(10) when referring to the rtv_order_no value. Please read the related documents for full details.

Stock Management

Defect 344346 describes that `stkupd.pc is not populating STAKE_SKU_LOC.IN_TRANSIT_AMT.` Please read the related documents for full details.

Defect 344495 describes that the insert into if_tran_data and the delete from tran_data statements are resulting in decreased performance. Please read the related documents for full details.

System Setup

Defect 344506 describes that global variables are not populated when forms are opened without going through rtkstrt. Please read the related documents for full details.

Instructions to reduce chaining

Applies to tables

item_loc_soh
tsfdetail

DDL Changes:

Added the following columns to item_loc_soh

Column	Type	Initial Value	Updated when
FIRST_RECEIVED	DATE	NULL	First PO Receipt for item/loc
LAST_RECEIVED	DATE	NULL	First PO Receipt for item/loc
QTY_RECEIVED	NUMBER(12,4)	NULL	First PO Receipt for item/loc

Added the following column to tsfdetail

Column	Type	Initial Value
PUBLISH_IND	VARCHAR2(1)	‘N’ or ‘Y’

Description

Chaining occurs when a row on a database table cannot fit on just one block. When this happens, a pointer is placed in the block. The pointer points to a linked list of chained records. Accessing this linked list can impact performance dramatically. Depending on PCTFREE for the tables affected, you may or may not experience chaining after the patch installation.

Note that the PCTFREE on item_loc_soh is automatically adjusted in the patch to 25. This is because the initial values for the new columns will be NULL, and will be updated with values over time. We recommend analyzing this table for chaining on a regular basis. If a large amount of Purchase Order receiving is being done in the system, we recommend doing an analysis every three months.

Instructions

1 Run the following two commands in SQL

   a ANALYZE TABLE ‘<TABLE_NAME>’ ESTIMATE STATISTICS SAMPLE 1 PERCENT

   b SELECT NUM_ROWS, CHAIN_COUNT, CHAIN_COUNT/NUM_ROWS CHAIN_PCT FROM USER_TABLES WHERE TABLE_NAME = ‘<TABLE_NAME>’

2 Look at the result in the column CHAIN_PCT

   If CHAIN_PCT > .01, consider using standard techniques to clean up chained rows.

   If CHAIN_PCT > 1, we recommend using export/import to eliminate chaining.

The following document about chaining is from Oracle Metalink.

- Goal: How to Diagnose and Solve Row Migration and Row Chaining?
- Fact: Oracle Applications 11.5.5 (Application object library)

Fix

What is Row Chaining?

The row is too large to fit into an EMPTY data block. In this Oracle stores the DATA for the row in a CHAIN of one or more Data BLOCKS. CHAINING occurs when row is INSERTED or UPDATED. Row chaining can happen for very large rows such as rows that contain LOB. Row chaining in such cases is Unavoidable.

What is Row Migration?

An UPDATE statement increases the amount of DATA in a ROW so that the Row NO LONGER FITS in to its DATA BLOCK. Oracle tries to find another Block with enough free space to hold the entire row. If such block is available Oracle moves the entire ROW to the NEW BLOCK. Oracle keeps the original Row piece of a Migrated row to POINT to the NEW BLOCK containing the actual row. The ROWID of the MIGRATED rows does not change. INDEXES are not updated and they point to the ORIGINAL row LOCATION.

Understanding

Migration and Chaining have negative effects on performance. INSERT and UPDATE statements that cause migration and chaining perform very poorly since due to additional PROCESSING. Queries that use an Index to select migrated or chained rows must perform additional I/O's. Migration is caused due to less PCTFREE set. There is not enough place in the BLOCK for UPDATES. To avoid migration all Tables that are updated should have there PCTFREE set in such a way that there is enough space within the BLOCKS for UPDATES.

How to Identify Row Chaining and row Migration

We can detect the Migrated or chained rows in a TABLE or CLUSTER by using the following methods. Using ANALYZE,REPORT.TXT and V$VIEWS.

Analyze

Before doing this Analyze create the table that can hold chained rows. Execute UTLCHAIN.SQL script found in the ($ORACLE_HOME)/rdbms/admin directory, this can be run to create the CHAINED_ROWS table or else create a simillar table with same column and datatypes and sizes as the CHAINED_ROWS table. SQL>ANALYZE TABLE SCHEMA_NAME.TABLE_NAME LIST CHAINED ROWS; SQL>SLECT owner_name,table_name,head_rowid from chained_rows where TABLE_NAME ='YOUR_TABLE_NAME';

REPORT.TXT and V$VIEWS:

The Migrated or chained rows can be detected by checking the "table fetch continued row" statistic in V$SYSSTAT or in REPORT.TXT. Note: REPORT.TXT is a file created during UTLBSTAT and UTLESTAT analyze.

How to Eliminate Migrated and Chained Rows:

It is VERY important to understand the distinction between migrated and chained rows. Row chaining is UNAVOIDABLE but row migration can be MANAGED and resolved through reorganization.

Note:

Chaining if it’s going to be in all Tables then it becomes a design issue and properly the DBlock size is done.

To avoid row chaining for very few occasions if you increase the DBlock size the effects of it as follows.

For Example:

Space in the Buffer cache will be wasted if you are doing random access to small rows and have a large block size. For a Block size of 8 KB and a 50 byte row size, the wasted space will be 7950 bytes in the buffer cache in an random access.

The steps invloved are:

1 Analyze the table ....list chained rows.

2 Copy the rows to another table.

3 Delete the rows from the original table.

4 Insert the rows from step 2 back to original table.

Step 4 eliminates the Migrated rows because Migration only occurs during an UPDATE. The migrated rows can be cleaned up with SQL plus script. Regular pruning of chained rows will improve the performance of your I/O subsystem. Increase the PCTFREE to avoid migrated rows.

.