Quantum Utility Quick Guide

Contents

[Overview 3](#_Toc27388840)

[Main Features 6](#_Toc27388841)

[Other Options 7](#_Toc27388842)

[Suggested Future Modifications 7](#_Toc27388843)

[Issues 8](#_Toc27388844)

[Questions 9](#_Toc27388845)

[Referenced Documents 9](#_Toc27388846)

[Code change record log 11](#_Toc27388847)

# Overview

Quantum Utility is a command-line utility that automatically FTPs files down from the UPS server for the current day and backs them to a local AWS drive (\\172.31.251.161\sdixdata\ftp\). The Windows Task Scheduler should run the executable on a daily basis at 11:15pm (to catch all daily exports provided previously by hour). The utility then emails users, whose contact information is accessible. Otherwise, results are also searchable via a QuantumView search page stored in the OnsiteOnline SDIX\SDIXZ folder.

Note: Users should remember to clear out the sdixdata\ftp folder archives every 1-3 months to avoid running out of space.

#### QuantumView Overview

UPS generates numerous plain text files per day/per hour containing shipping information, unique tracking numbers and variations of the SDI purchase order number. UPS could generate anywhere from 3-12 files daily (added throughout the day). The utility searches each file for records of type:

* E1 - Exception (packages that indicate any shipping issues),
* D1 -Delivery Short (basic delivery information provided a couple hours after delivery) and
* D2 – Delivery Long (records that provide more complete information).

Based on the provided plain-text tracking number and variations of the purchase order id the utility then searches the SDI database for matches on the following fields:

| Field Name | UPS QuantumView Field(s) | SDI DatabaseTable.Field(s) | Description\Mapping Logic |
| --- | --- | --- | --- |
| Tracking Number | TrackingNumber | PS\_ISA\_ASN\_SHIPPED.ISA\_ASN\_TRACK\_NO | This table is only 1 of 2 PeopleSoft tables that the SDIExchange id can access. M. Randall suggested using the PS\_ISA\_ASN\_Shipped table for potential Tracking Number matches.  Note: 90-100% of the time the tracking number are not stored in the SDI database when we receive the files. The only match M. Randall found in the DB was on 1Z1639370312224053 in the QVD\_ALT\_SDIINC\_20091119\_150122\_223\_SDIQVD.txt file.  A Purchase Order could have multiple line entries. Each line entry will have its own tracking number. Using the Tracking number as a PK, developer can pull the affiliated line number and PO ID from the database. If Tracking number does not match, users must use the PO\_ID provided by UPS. |
| Purchase Order ID | ShipmentReferenceNumber Values 1&2 and PackageReferenceNumber Values 1&2 | PS\_PO\_LINE\_SHIP.PO\_ID or  PS\_PO\_HDR.PO\_ID | The SDI ID is typically around 10 characters (the older versions were purely numbers, the newer versions start with 1-2 alphabetic characters and end with digits.  Initially, the code searched PS\_PO\_HDR which takes less time to search (initially recommended by M. Randall) but shifted to PS\_PO\_LINE\_SHIP which takes more time (upon noticing the InsiteOnline3.1 codebase (CommentDetails.aspx.vb or repostatus.aspx.vb ) use that table in their queries  UPS stores variations of the PO ID in 4 different fields. Sometimes just in ShipmentReferenceNumberValue2 or PackageReferenceNumberValue1, sometimes both ShipmentReferenceNumber1 and PackageReferenceNUmberValue2 (see QVD\_ALT\_sdiinc\_20191125\_170154\_826\_SDIQVD.txt), sometimes all 4 and sometimes only 1. As a result, the query searches each.  Additionally, UPS stores PO ID in the UPS Reference Fields sometimes with spaces, special characters (‘PO.R010115686’), or extra characters in front (‘PO#: ‘ , ‘SDI PO GR02857913‘), but most times as the standard SDI POID format (PM00068987). See QVD\_ALT\_SDIINC\_20191119\_120127\_313 for example variations.  The utility code runs through, strips the known special characters and attempts to match the possible variations to the PO\_ID in the SDI DB. Dev Team will need to update the QuantumUtility.StripChars() function if encountering new variations.  Most times a match is made on PO ID. Unfortunately, UPS typically does not provide line number, meaning if the tracking number does not exist in the SDI DB, there is no way to know which purchase order item the entry belongs to. |

Once the codebase matches on PO ID (and/or Tracking Number), it adds a new line to the SDIX\_UPS\_QUANTUMVIEW\_LOG table.

If the utility does or does not find any match on the tracking number or purchase order id, an alert is written to the SDIX\_UPS\_QUANTUMVIEW\_ERROR table as well.

Code errors are written to the SDIX\_UPS\_QUANTUMVIEW\_ERROR table. Also, if a file does not contain D1, E1 or D2 entries, it is skipped entirely. A corresponding message is written to the Quantum View Error table. A running log file for the day is written to the corresponding AWS folder. To turn it off just comment out the LogErrorFile lines.

#### code installation

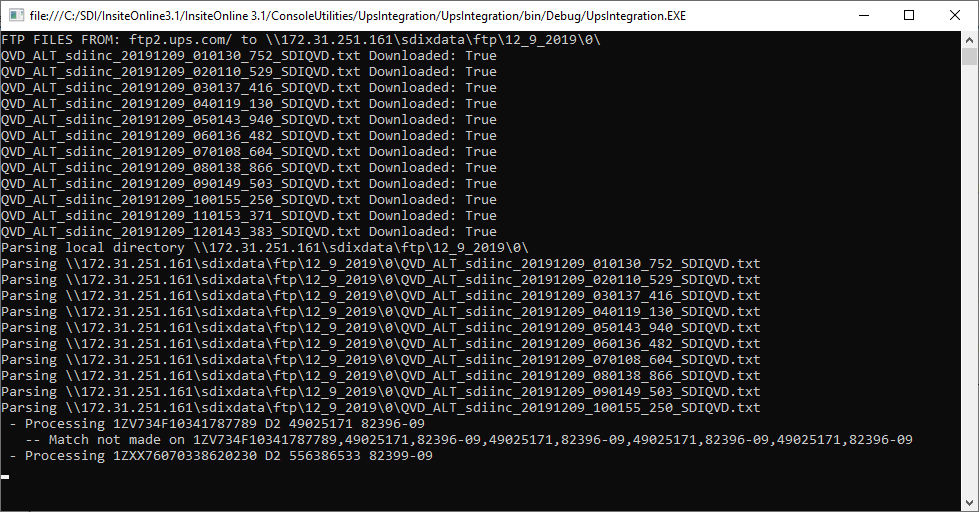
Users should install the entire UPSIntegration package on the server (which contains the necessary WinSCP FTP and Quantum View support utility files).

At the command line, users can either type UpsIntegration.exe (or UpsIntegration) to run that day’s download or UpsIntegration.exe dd//mm//yyyy to pull down for a specific day i.e. UpsIntegration 12/04/2019. Note: users must use that specific date format. Otherwise, the application defaults to current day.

The utility quickly parses each file row until it finds an E1, D1 or D2 match and then verifies if the POID/TrackNum exists in the DB. All search results are output to the console, SDIX Quantum LOG table and local c:\temp\error file.

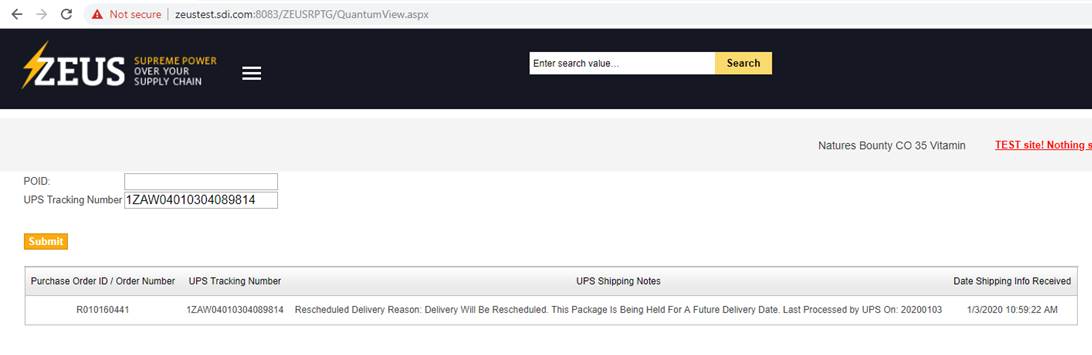
If a file does not contain the E1, D1, or D2 type, the application skips it entirely and pushes an alert to the SDIX Quantum Error table.

Backed up files are saved to AWS (\\172.31.251.161\sdixdata\ftp\) by the date the application is run and stored in a subfolder containing the files pulled down for the inputted date i.e. \\172.31.251.161\sdixdata\ftp\12\_12\_2019\12\_5\_2019\



#### viewing RESULTS

Users can navigate to the QuantumView.aspx page. Users must have the exact POID or tracking number (to avoid displaying information that does not belong to them). This searches the table that imports all of the UPS Quantum View data.

Note: While most users (at least 50%) should receive an email alert, those that do not (if their purchase order is not stored in PS\_ISA\_ORD\_INTF\_LN) can still search the page to receive their update. 

#### When to run

UPS updates their FTP files throughout the day. Scheduled job should run at 11:15PM EST daily.

#### System Debugging

The application displays the application progress on the command line. However, it also writes the results of the matches (and parameters) to the SDIX\_UPS\_QUANTUMVIEW\_LOG. If there are any system or C# errors, the raw errors are written to the SDIX\_UPS\_QUANTUMVIEW\_ERROR table. There is also a log file written to the corresponding AWS folder.

## known issues

* 0.5% Tracking ID Match: Almost none of the tracking ids provided by UPS are in any of the SDIX tables
* Partial POID Match: UPS does not always provide a valid POID. When UPS provides POIDs, sometimes they are improperly formatted or do not exist in the relevant SDIX tables.
* Partial User Email Match: Only 50% of the POIDs have corresponding entries in the purchase line table (which provides end user information)
* Processing Speed: Each CSV row takes 15 to 60 seconds processing time. A CSV file with 5 E1 entries, will take around 3-5 minutes processing time. A CSV file with 119 E1 entries might take 1-2 hours to process. A folder with 20 files (5 of which contain over 100 D1/D2/E1 entries) could take 5-7 processing hours. Although simplifying the query joins improved the speed, we could not take advantage of the contemporary CSVHelper parsing package (as not compatible with VS12).
* Quantum Search: Added the quantum view search page, so that users who did not receive emails (but had tracking ids or poids) could at least search the LOG table for the provided data. Does not provide for partial search to avoid users pulling data not relevant to their IDs. Checked into SDIX\SDIXZ solution.
* UPX QuantumView file can have redundant entries -   
  Sometimes QuantumView files can have over 20 (and up to 100) redundant tracking id and poid entries with the exact same row data. In response, decided to store redundant data in LOG table, but will only email users the FIRST entry of the redundant file

## Versions

Iteration 1: AVASOFT created test code that parsed a hard coded comma-separated CSV file into its own table. Avasoft eventually removed the table after the project went on pause.

Iteration 2: New Developer created standalone utility that downloaded all the files, parsed the data into its own table (using POID and Tracking ID) and also updated PS\_AS\_XPD\_COMMENT Notes\_1000 field (based on M. Rhandall suggestion), where users could view the comments history via the UI by navigating to the Expedite PO Update Page, searching for that specific PO ID and clicking the comments history icon. Note: Because the Comments table uses PO ID and Line Number as a combination key, possibility of users not seeing their data (if the code cannot find a line number match) removed the option, especially as comment history ASPX page pulling from the PS\_AS\_XPD\_COMMENT table required both a line number and POID as a composite id.

Iteration 3:. After talking to key users, developer updated the email component and added QuantumView.aspx search page in SDIExchange (SDIX/SDIXZ, InsiteOnline.sln ). Tried adding latest CSVHelper but removed as C# 6.0 shorthand not supported in VS12.

## Main Features

Below find key application features:

* Searches on TRACKING ID and PO ID: Because the tracking number is not always found in the shipping table, it also searches the relevant Purchase order tables.
* Processes pipe-delimited and Tab-delimited files: Most of the text files on the UPS server use the pipe (|) character to separate data. However, the QVD\_ALT\_sdiinc\_20191115\_110106\_627\_SDIQVD.txt contained tab-delimited characters. It is possible a user accidentally re-saved it with tabs. But, just in case, the application also handles tab-delimited files.   
  Note: Because white space in Tab-delimited is handled differently when read into a C# string buffer, the code searches for the tracking number and purchase order ids using regular expression patterns.
* Handles Multiple field Arrangements: The file fields might not appear in the same order. Additionally, not all the fields will always appear in each file. Additionally, M. Randall and other key SDI users can request UPS adjust the data included in the outputted text files. As a result, the utility does not assume each file will contain the same fields in the same order. It runs through the header of each file to identify the contained fields. On a bonus, the field header titles never vary and always stick to camel case i.e. RecordType/ShipperNumber – Hence the reason a ToUpper search is not used.
* Pulls from 4 different fields to identify purchase order: Purchase order does not always appear in the files, in the same format or in the same fields. Because the SDI PO ID typically contains 10 characters. Older PO IDs were numeric. Newer ones typically start with 1-2 letters, followed by numbers. The PO Ids in the QuantumView tracker text files could appear in the ShipmentReferenceNumberValue fields (1&2) and/or the PackageReferenceNumberValue fields (1&2). Additionally the PO ID appearing in the UPS file might contain a dash(-), colon( : ), pound sign(#), extra spaces or extraneous letters.   
  Sample Values contained in actual files include: TX57-00075945, PM00069193, 4507699387, PO# 8044-1114, PO 0180214017, OH41-00289460, PO L04E094949, L01E094238, N22913361-00-001 PICK  
  The utility tries the values found in all 4 fields with 1-2 different variations per field (removing special characters) to find a match.
* Backs up files to AWS Server: [\\172.31.251.161\sdixdata\ftp](file:///\\172.31.251.161\sdixdata\ftp\12_12_2019\12_5_2019\)
* if run on the same day with the same parameters, the utility deletes pre-existing files - If user FTPs files on 12092019 for 12092019, the utility will create a 12092019/12092019 folder. If the utility is re-run again on 12092019 for 12092019, the utility will pull down fresh files (and delete the earlier ones from the local). However, if users process files on 12/09/2019 from 5 days ago, the utility will not delete the earlier 12092019 and just create a folder 12092019/12042019
* pull files for a specific date. Currently, app defaults to current date. But, users can specify a previous date by entering DD/MM/YYYY format. If date format is incorrect, app will pull down current day’s files. If date is greater than 15-19 days, app will warn data might not exist on UPS server.
* warns users that files might not exist on server if date entered over 15-19 days old
* Emails Users
* write log files to AWS directory

But, can also save them to a local c:\temp\error – Both are through the QuantumUtility.writeErrorFile capability

## Other Options

The below items are options coded into the system but would require tweaking to activate -

* FTP files between specific date ranges (Exists in QuantumView and FTP in QuantumUtility)

## Suggested Future Modifications

The following items are suggested for the future to ideally speed up queries and make data responses more immediate –

* Use Stored procedures instead of database selects
* Use Quantum API instead of FTP
* Alert if files not FTPd to local (base functionality already exists, just need to activate)

## past Issues

The below are past identified data issues :

* Communication table OPERATOR ID – Previously, all Com entries entered from the application use operator id (OPRID) SDIX or SDISOLUT for troubleshooting ease.
* Query Timeouts: Long queries encapsulating triple joins which took longer than 47 seconds in SQL Developer tended to timeout. Increased DB connection time decreased the timeout. However, as code no longer writes to the COM HISTORY table, removing that drastically improved query speed to 1.75 seconds.
* None of the tracking #s were found in the relevant RPTG /STAR database tables

As none of the tracking numbers were matched, only the POIDs (50% of the time), it was decided to search in the PO tables, as well. And, would email users based on the POID [\\172.31.251.161\sdixdata\ftp\12\_10\_2019\12\_5\_2019\QVD\_ALT\_sdiinc\_20191205\_010147\_189\_SDIQVD.txt](file:///\\172.31.251.161\sdixdata\ftp\12_10_2019\12_5_2019\QVD_ALT_sdiinc_20191205_010147_189_SDIQVD.txt)

* no way to identify corresponding SDIX line nbr ID   
  Each purchase order can have numerous items ordered. Each item ordered will have a unique line number on the purchase order i.e. 1, 2, 3. For data to appear in the ui, most queries, including the page that displays COM history, require a POID and LINE NBR. As a result, decided against inserting into COM history table to avoid redundant entries in COM table.
  + Redundant entries in COM table, currently the same POID might have multiple tracking numbers in the same file. That means that POID might have multiple entries in the comments table
  + matchES on POID, but not line\_nbr Initially, the program tried using the last line number in the DB for that PO ID, then it tried using 0s. But, assumedly, a pre-existing unique POID/Line NBR/Sched Nbr combination is required.
* Some of The POIDs matched in PS\_POHDR might not appear in SHIPPING either
* UPS alternates the PO ID format. There are \_MANY\_ special character variations for the POID provided by UPS. I currently update the code everytime I encounter a new variation, but future developers should check the LOG table every now and again to update the code accordingly
* What happens when a POID has multiple line\_nbrs After reviewing the CSV files, I attempted to identify a way to match a POID in a UPS Quantum file to an entry in a PO table using weight/etc. but that did not seem viable. Future developers could review that option.
* For inserting into the ps\_isa\_xpd\_comment table, utility uses the default ‘SH’ as the problem code.
* Other Data Discrepancies – Sometimes Tracking number appears twice, both in tracking number field and the value fields but with spaces i.e. 1Z9161990348652126,1Z 916 199 03 4850 8710,,1Z 916 199 03 4850 8710,,1Z 916 199 03 4850 8710 as seen in [\\172.31.251.161\sdixdata\ftp\12\_16\_2019\12\_3\_2019\QVD\_ALT\_sdiinc\_20191203\_010145\_585\_SDIQVD.txt](file:///\\172.31.251.161\sdixdata\ftp\12_16_2019\12_3_2019\QVD_ALT_sdiinc_20191203_010145_585_SDIQVD.txt)
* Sometimes it hangs on files greater than 100 lines or when DB connection slow  
  It will keep processing, but the below file contains 123 lines and hung around line 52 or 53, right before 1Z721YY00361420759, for 5 min, but then continued processing normally [\\172.31.251.161\sdixdata\ftp\12\_17\_2019\12\_17\_2019\QVD\_ALT\_sdiinc\_20191217\_130126\_240\_SDIQVD.txt](file:///\\172.31.251.161\sdixdata\ftp\12_17_2019\12_17_2019\QVD_ALT_sdiinc_20191217_130126_240_SDIQVD.txt)
* data in same file might not have same data in db  
  For instance, in files downloaded for [\\172.31.251.161\sdixdata\ftp\12\_17\_2019\12\_17\_2019\](file:///\\172.31.251.161\sdixdata\ftp\12_17_2019\12_17_2019\), I found entries for the req\_id\order\_no (matching the Po\_id in ps\_po\_hdr) in PS\_ISA\_ORD\_INTF\_LN for (req\_id\order\_no = 'SG00504353' or req\_id\order\_no='NY00492986') corresponding to the poids (po\_id='S010504353' or po\_id='N010492986')  
  However, po\_id='AM00175758' which had a corresponding req\_id\order\_no in ps\_po\_line\_distrib for req\_id/ (order\_no = '0000215866'), did not have corresponding data in the PS\_ISA\_ORD\_INTF\_LN table

## PAST Questions

* Are there any other fields you might want to pull from the files? According to power user Don, after forwarding on quick view guide and raw csv file (alongside sample excel), no.
* Sometimes POIds have 2 values in one – initially the utility did not parse them out–

0290214800/00315055042 OR 335520065171741, 8593793254 OR AS04041816 & AS04040762,S-14341 & S-14342 – Should these be handled separately? Don forwarded on the question, but after manually breaking apart those digits and searching the relevant PS\_PO\_HDR table in the RPTG db, no match was found. Also, more recent PO IDS are alphanumeric as opposed to purely numeric. So, the code will continue to search the db for 0290214800/00315055042 but it will not break them up by slash. However, the code is still there to use (that can recusively call the parserow code).

* Testing Multiple Database Options: Use RPTG to test data rather than synching PS\_PO\_HDR, PS\_ISA\_ASN\_SHIPPED and PS\_AS\_XPD\_COMMENT tables from RPTG to STAR (after discussion with Rashmi and M. Randall)
* Redundant Entries*:*  After talking to M. Randall, even if the same POID might have multiple tracking numbers in the same file, considering the notes\_1000 field is only 1000 characters wide, the utility will have to create multiple entries for each POID with a unique tracking number in the comments table to appear in history. For instance, POID C01G591513 had 8 unique tracking numbers in the [\\172.31.251.161\sdixdata\ftp\12\_10\_2019\12\_5\_2019\QVD\_ALT\_sdiinc\_20191205\_010147\_189\_SDIQVD.txt](file:///\\172.31.251.161\sdixdata\ftp\12_10_2019\12_5_2019\QVD_ALT_sdiinc_20191205_010147_189_SDIQVD.txt) *Talked to end user about lack of line number to create composite key with PO ID. Currently on pause*
* For emailing, currently, users want to email vendor directly rather than batching. Will review DB export provided by Wenjia and work with M. Randall to identify tables containing vendor emails and will review older code for SMTP data. Sometimes Vendor\_Id contains vendor email directly. After all of the files are processed, a separate method should exist called before DB connection is closed that runs through, finds all tracking numbers for each vendor (rather than sending vendor 20 emails, they’ll just receive one). Then, if users decide to batch emails, they can pull this method out as a standalone.
* Using SDIX as fake operatorid When there’s a match on POID, but there’s no corresponding line\_nbr, Sched\_nbr or oprid, decided to use something other than 0s for entry into the com table. *OPRID is operator ID, so created test id called “SDIX’, Or pre-existing employee id SDISOLUT. This will uniquely identify entries in the UPS exports generated by utility. Talked to M. Randall about creating fake id in users table, if necessary.*
* Use SH as ISA\_PROBLEM\_CODE Used to use record type from quantum extract*. However, After reviewing vb code and db entries, noticed the initial PO\_ID entries use AK i.e. Acknowledgement. But, since this information relates to Shipping should use the ISA\_PROBLEM\_CODE ‘SH.’ Need to use the pre-existing ISA\_Problem\_codes or data will not appear in the UI comment history as they query checks the table that holds the ISA\_PROBLEM\_CODE affiliations.’*
* RIght now, don’t create dummy ids if no matching POID/Line NBRs found as the POID/LINE\_NBRS composite id correspond to actual invoice lines on a specific PO.

## Referenced Documents

* Code Documentation – See attached excel. This application contains 3 utility files for FTP, Database and string manipulations, in addition to the code
* CSV\_File\_guide\_v8 – Jan2016.pdf For more info on the formatting of the plain text Quantum View files, please see (provided by M. Randall)
* UPS Integration 1.2 and 2.2 – For a high level, first pass requirements document please see, (provided by M. Randall V. Karthikeyan, whose team created the first pass program.cs)

## Code change record log

Below find key code changes in the past three weeks upon discovering new data requirements or new tables:

|  |
| --- |
| New query on 11/26 based on new comments from M. Randall  Switched to using isa\_asn\_shipped for tracking number and searching ps\_po\_hdr for po\_id  12 switching to ps\_po\_line which has more data than ps\_po\_hdr |
| New Query change made on 12/16  Added join on PS\_PO\_HDR as sometimes what appears in PS\_PO\_HDR does not appear in PS\_PO\_LINE\_SHIP. While PS\_PO\_HDR is preferred table recommended by M.Randall, PS\_PO\_LINE\_SHIP contains more PO related data i.e. line nbr, etc. |
| New query on week of 12/11 based on review of NeedApprove.aspx.vb and VendorProfile.Aspx.vb  Switched to using isa\_asn\_shipped for tracking number and searching ps\_po\_hdr for po\_id  12 switching to ps\_po\_line which has more data than ps\_po\_hdr |
| Added ability to parse on tabs and pipes week of 12/5  Parsed file with tabs that generated errors. Added check to see if separator = ‘|’ appears in CSV file, if yes, use standard processing, if not, use tab delimited processing which searches for the POID/Tracking number pattern |
| New Query Week of 12/5  Checked PS\_PO\_LINE\_SHIP temporarily from PS\_PO\_HDR as PS\_PO\_LINE\_SHIPPED contained more information |
| New Query Week of 12/10  Temporarily Used PS\_APXVCHRLINE tbl until noticed the reportstatus.aspx.vb used PS\_PO\_LINE\_SHIP |
| Query Update Week of 12/11  Updated the comFromSql line – Previously it looked like the below. Updated the LINE NBR, SCHED NBR restrictions as those are typically empty on the shipping table side (as shipping table typically does not have a data match on tracking number)  //String comFromSql = “ LEFT JOIN PS\_ISA\_XPD\_COMMENT COM ON PO.PO\_ID = COM.PO\_ID AND COM.BUSINESS\_UNIT = SH.BUSINESS\_UNIT AND COM.LINE\_NBR=SH.LINE\_NBR AND COM.SCHED\_NBR=SH.SCHED\_NBR AND COM.OPRID=SH.OPRID “; |
| Week of 12/9 and 12/5 – Added then removed handling possibility of nested POID logic  Initially added nesting logic as some PackageReferenceNumberValue fields contain numerous digits separated by slashes or commas. Wanted to check to see if they were POIds. Checking them manually, none of them were. Also, running this code caused stack overflow, if recursively calling parseRow 2-3 times while it’s still parsing a current row. Might require closer debugging to tighten code logic and eliminate stack overflow possibility.  /\* // if package value ref 1 contains a field separated by ?, & or , - then split up into separate poids  if (qf.PackageReferenceNumberValue1.Contains(“,”) )  {  String[] poid = qf.PackageReferenceNumberValue1.Split(‘,’);  foreach(var po in poid)  parseRow(currentRow, qf, row, filename , po );  }  if (qf.PackageReferenceNumberValue1.Contains(“&”))  {  String[] poid = qf.PackageReferenceNumberValue1.Split(‘,’);  foreach (var po in poid)  parseRow(currentRow, qf, row, filename, po );  }  if (qf.PackageReferenceNumberValue1.Contains(“/”))  {  String[] poid = qf.PackageReferenceNumberValue1.Split(‘/’);  foreach (var po in poid)  parseRow(currentRow, qf, row, filename, po);  }\*/ |
| Updated query logic – week of 12/11 and 12/5   * Initially, data only did query on one default database connection. But, for testing purposes realized STAR was not up to date, so did query on first star then RPTG and if matches existed in STAR, then updated RPTG. Learned later on that RPTG is another dev DB just cleared out daily so could use that. Updated logic back to using only one default DB. * Also, initially SQL first checked PS\_PO\_HDR on PO\_ID and then ISA\_ASN\_SHIPPED table on tracking number to identify amount of matches made on PO ID vs Tracking number. But, after no matches found on tracking number, just reverted to using PO ID (but kept the Shipping table left join)   /\* No longer doing separate check of SH table as main table – since not finding on Track num, but will leave in left join This existed just to see which table was matched and 100% it was PS\_PO\_HDR on PO\_ID and not track # \*/  if (!dbReader.HasRows || dbReader == null)  {  dbReader = QuantumDbUtility.executeDbReader(dbConn, asnSelectSql + shFromSql + comFromSql + whereSql, dbParams);  sdix\_ups\_quantumview\_log\_params[5] = “TRUE: ISA\_ASN\_SHIPPED”;  } |
| Updated logic for inserting into COM table on 12/10  After further discussion with m. Randall on 12/10 since there is a log history, will no longer update, just insert  \* also will just use isa problem code for SH \*/ |
| Updated Insert Query to COM table to include current timestamp week of 12/15  Note: Query also pulled the previous OPERATOR, but in the end for any new entries will use OPRID, SDIX  ps\_isa\_xpd\_comment\_params = new String[8] { qf.business\_unit, qf.ps\_po\_id, qf.ps\_line\_nbr, qf.ps\_sched\_nbr,  qf.isa\_problem\_code , qf.ps\_notes\_1000\_new, “SDIX” /\* QuantumUtility.returnNull(dbReader[“SH\_OPRID”].ToString()) \*/,  /\*”31-DEC-1999” \*/ DateTime.Today.ToString(“dd-MMM-yy”).ToUpper() }; |
| /\*  if (!String.IsNullOrEmpty(qf.ps\_notes\_1000)) // && updateStar  {  QuantumDbUtility.executeDbUpdate(dbConn, “Update PS\_ISA\_XPD\_COMMENT SET NOTES\_1000= ‘@5’ WHERE PO\_ID=’@1’ and “ + “ BUSINESS\_UNIT=’@0’ and LINE\_NBR=@2 and SCHED\_NBR=@3 and OPRID=’@6’ “, ps\_isa\_xpd\_comment\_params);  QuantumUtility.logError(“ -- Updated PS\_ISA\_XPD\_COMMENT.NOTES\_1000, PO: “ + qf.ps\_po\_id); //indicate num of rows updated sdix\_ups\_quantumview\_log\_params[3] = “Updated PS\_ISA\_XPD\_COMMENT.NOTES\_1000-PARAMS: “ + String.Join(“,”, ps\_isa\_xpd\_comment\_params);  QuantumDbUtility.executeDbUpdate(dbConn, sdix\_ups\_quantumview\_log\_sql, sdix\_ups\_quantumview\_log\_params);  }  else \*/ |
| Defaulted to using ‘SH’ which stands for Shipping for ISA\_PROBLEM\_CODE in the COM table  Especially since a) difficult to match on line number b) this should be a new entry that will appear in comments history. Currently test data does not appear in UI without line number match, we could do a comments history on the overarching PO page if internal client and web services mgr amenable – Internal, client will verify if they want to see tracking data if can’t match to line number  //ps\_isa\_xpd\_comment\_params[4] = qf.RecordType;  //\*\*\* WHAT DO I USE FOR ISA\_PROBLEM\_CODE IF IT IS AN INSERT? What happens when there’s a match on POID=’AM00173902’ in ps\_po\_hdr,but there’s no corresponding line\_nbr, Sched\_nbr or oprid from the ship table? Use 0s? verify before inserting into notes row doesn’t already exist verify the test data appears in the UI |
| Updated log table  Create table SDIX\_UPS\_QUANTUMVIEW\_LOG (  /\* use if ORA 11 – ups\_quantumview\_id number(11,0) NOT NULL constraint pk\_ups\_quantumview\_id primary key, \*/  /\* Use if ORA12C or higher – ups\_quantumview\_id number GENERATED ALWAYS AS IDENTITY, \*/  ups\_filename varchar2(500) null,  po\_id NVARCHAR2(250) null,  isa\_asn\_track\_no VARCHAR2(30) null,  ups\_sdi\_match varchar2(45) null,  utility\_action varchar2(250) null,  ups\_file\_location varchar2(500) null,  dttm\_added TIMESTAMP DEFAULT CURRENT\_TIMESTAMP  );  /\* Use triggers for auto increment, if Oracle version is Oracle11  CREATE SEQUENCE sdix\_seq\_ups\_quantumview\_id START WITH 0;  CREATE OR REPLACE TRIGGER sdix\_trgr\_ups\_quantumview\_id  AFTER INSERT ON SDIX\_UPS\_QUANTUMVIEW\_LOG FOR EACH ROW  BEGIN  SELECT sdix\_seq\_ups\_quantumview\_id.nextval INTO :new.ups\_quantumview\_id FROM dual;  END;  \*/  Create table SDIX\_UPS\_QUANTUMVIEW\_ERROR (  /\* use if ORA 11 – ups\_quantumview\_error\_id number(11,0) NOT NULL constraint pk\_ups\_quantumview\_id primary key, \*/  /\* Use if ORA12C or higher – ups\_quantumview\_error\_id number GENERATED ALWAYS AS IDENTITY, \*/  error varchar2(1000) null,  dttm\_added TIMESTAMP DEFAULT CURRENT\_TIMESTAMP  ); |
| Based on the provided plain-text tracking number (typically 18-characters) and variations of the purchase order id found in the ShipmentReferenceNumber (Values 1&2) and PackageReferenceNumber (values 1 &2) fields, the utility searches the SDI database for matches. |
| Testing DB  Per conversation with Rashmi and M. Randall, suggested not to create test entries in the PS tables on STAR (or back up some of the RPTG tables to START) but to test against Reporting directly . Also RPTG has the more recent POIDs sent over from UPS |
| 12/17 Updated OPRID, SDIX, to existing employeeid, SDISOLUT |
| 12/18 Started integrating Email Capability  While testing the QUantumEmailUtility from local laptop received SMTPException: authentication failed because remote party closed the transport stream . Yury recommended using the SDIEmailUtility Service which the newer codebase uses. I added <http://ims.sdi.com:8913/SDIEmailSvc/EmailServices.asmx> as a service reference and used the base code provided  SDiEmailUtilityService.EmailServices SDIEmailService = new SDiEmailUtilityService.EmailServices();  SDIEmailService.EmailUtilityServices("Mail", "[SDIExchADMIN@sdi.com](mailto:SDIExchADMIN@sdi.com)", "[webdev@sdi.com;](mailto:webdev@sdi.com;)", testOrProd + "Error from ExpediterReload Utility", "", "", strBody, "SDIERRMAIL", MailAttachmentName, MailAttachmentbytes.ToArray());  To add a web reference in VS 2012, Right click on Service Reference, then hit advanced. Then, include the Web service link and and web service name ‘SDIEmailUtilityService’ – Declare Using [yourpackagename].SDIEmailUtilityService  If you aren’t using file attachments, just pass in null arrays |
| 12/20 Removed insert into comments history table Currently, inserting into comments table without verified line nbr could cause matches to unrelated data  Creating a new search screen instead (which is possibly what avasoft originally planned to do ) |
| Previously, the SDIX\_UPS\_QUANTUMVIEW\_LOG\_PARAMS – inserted the activity status into the LOG table Ie.e indicating if inserted new record into COMMENTS table or not (now, as no longer insert into comments table, this defaults to NOTHING) – Previously, we also tracked if the insert was true and what table it matched on i.e. PS\_PO\_LINE\_SHIP or other. Now, by default, PS\_PO\_LINE\_SHIP is the main table joined on so alternate queries are not run |
| 12/27-1/2 Updated email format (and corresponding db structure) based on client request. Also noticed one of the TEXT files could contain over 20-100 repetitive rows, so will track data, but only send client the first update |
| 1/2/2020 Removed Quantum email utility  Put into backup since an SDI email utility already exists |
| 1/2/2020  Increased connection timeout to 410 from 310 when 2 different 47 second query in SQL Developer timed out in c# |
| 1/8/2020 Updated original DB query  As we are no longer inserting into the notes table, don’t need to pull those fields – removed to speed up query. Query went from 47 sec to 1.675 sec. Note: Textually processing each line still takes time. I’m still leaving the initial select query in there. As UPS sends variations of the PO ID, we still need to verify WHICH of the poids included in the package and shipping reference values is valid. |
| Text Processing Speed *1/8/20*  Simplifying the query increased the line by line speed. However, it still takes at least 15 seconds per line to process. Although I attempted to use the CSVHelper package, recent versions were not compatible with VS12 and older versions required more work. See below.   * Did not use textreader as it did not appear to header processing without added manual manipulation. Also, as not all files have the same fields and not all fields appear in the same position, did not want to assume field[16] would always use the same data. * Tried using 12.2.1, the latest version of CSVHeader. However, it is built for C# 6.0 and the coding shorthand does not work with older VS versions under 2015 * Tried using slightly older versions from 2017(v3.0) and 2016 (v2.16 ) for CSV Helper . Unfortunately, it still uses the C# shorthand not compatible with VS 2012. * 2.0 and 2.1 are main versions that compile in VS2012 without tossing errors in VS 12. However, automapping did not take for either version. * For 2.1 created a map of the necessary fields. Unfortunately, still received error as seemingly need to map ALL the fields that appear in UPS file, so reverted back to manual coding |