* [Morning Batch Jobs](#MorningBatchJobs)
  + [Exports / Imports](#MorningBatchJobsImportExport)
    - ETB List
  + [FTP Routines](#MorningBatchJobsFTP)
* [After Trading Batch Jobs](#MorningBatchJobsAfterTradingHours)
* [Real-time processes](#RTP)
  + [Drop Copy Servers](#RTPDropCopies)
  + [P3 TradeProcessingEngine](#RTPP3)
  + [MH TradeProcessingEngine](#RTPMH)
  + [MarketDataRequestService](#RTPMDRS)
* [Excel Integrated Tools](#ExcelTools)
  + [DataCache](#ExcelTools)
  + [Analytics](#ExcelToolsAnalytics)
    - [Function Descriptions](#ExcelToolsAnalyticsFuncs)
  + [Misc](#ExcelToolsMisc)
    - [Ratio Sheet](#ExcelToolsMiscRatioSS)
* [Database](#Db)
  + [Used Tables](#DBUsed)
  + [Microhedge Database](#DBMH)
  + [Microhedge Risk](#DBMHRisk)
* [Process Locations](#ProcessLocations)
  + [Fifo\_pnl](#ProcessLocationsFIFO)
* [Expiration Cycle](#Expiration)
* [Misc](#Misc)
  + [MOC Algo](#MiscMOC)
  + [Docs, etc…](#MiscDocs)
  + [Code](#MiscCode)
* [Troubleshooting](#Troubleshooting)
  + [Non standard options entry / issues](#TroubleshootingNonStandardOptions)
  + [Position mismatch issues](#TroubleshootingPositionmismatch)
  + [Spreadsheet symbol not updating](#TroubleshootingSSSymbol)
  + [DropCopy not working](#TroubleshootingDropCopies)
  + [Entered wrong trade data into BrokerTrades Table](#TroubleshootingWrongBrokerEntry)
  + [Entered wrong trade data in ManualTrades table](#TroubleshootingWrongManualEntry)
  + [Historical Data cache not handling split information properly](#TroubleshootingHistorical)

**Morning Batch Jobs**

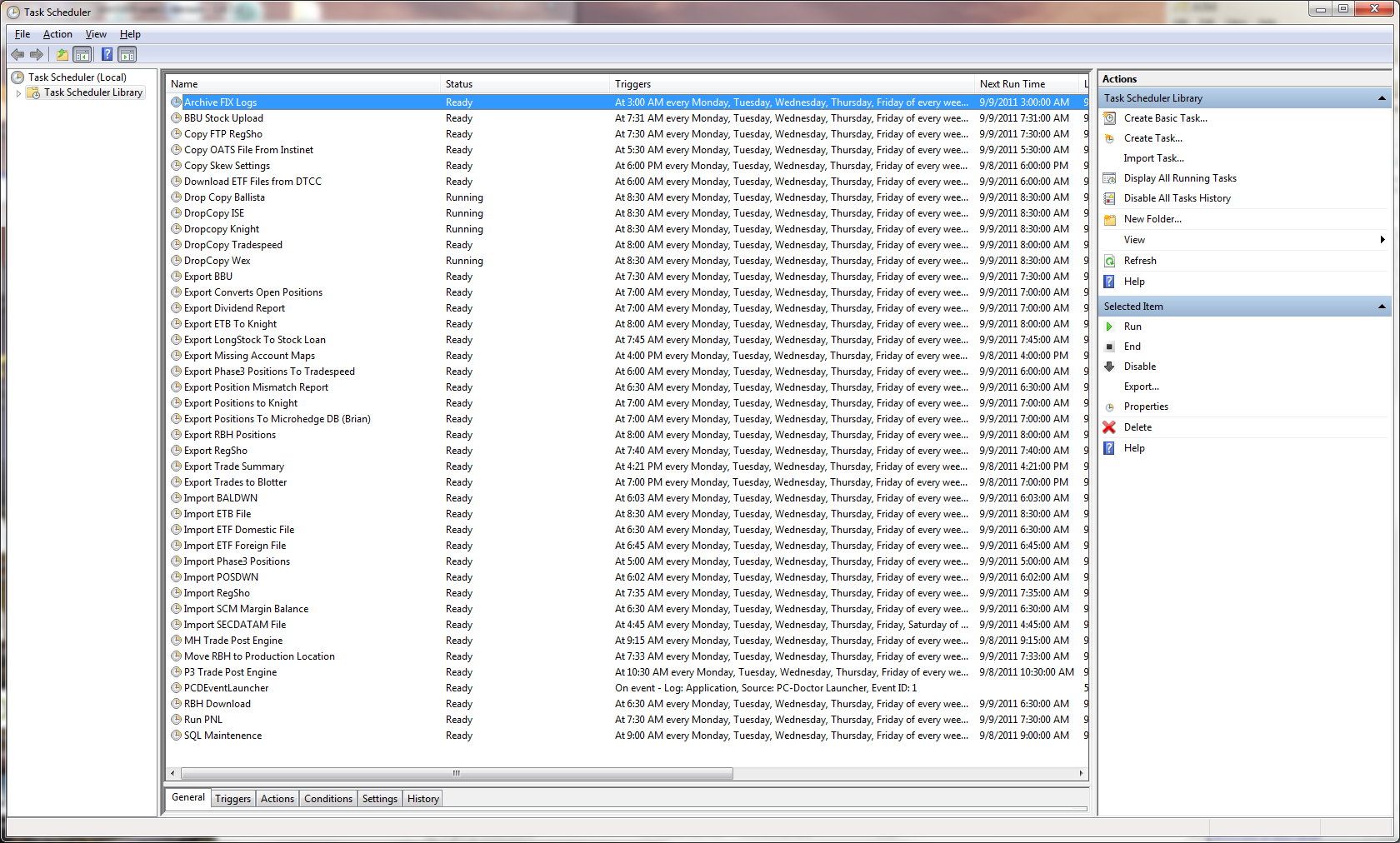
The morning batch jobs are a series of scheduled tasks used to download, import and export files after manipulating data in certain ways. The programs used in the batch jobs are located in **C:\Trading\Programs\Batch\bin** while the batch jobs are located in either **C:\Trading\Programs\Batch\Imports** or **C:\Trading\Programs\Batch\exports** or **C:\Trading\Programs\Batch\misc**

The description of the programs in the bin directory are:

* **ActivOptionsDailyRequest** (deprecated) – was used to pull options closing prices for the active positions in the NY81 account
* **Copyftp** – used to either download or upload files via ftp, or simply to move files from one location to another. Program uses command line parameters to specify actions similar to:
  + **copyftp -ftp-server iftp.instinet.com -ftp-uname AlbertFried -ftp-pwd Albert1222 -mailto rn@albertfried.com,akatsingris@albertfried.com -mailsvr mail -d down -ffile pub/AFCOyymmdd.txt -tfile \\mail\User\_Share\tradespeed\newoats\AFCOyymmdd.txt -t 1**
* **FileExporter, FileImporter** – these are generic engines used to load a class adhering to a specific interface and execute the classes ExportData or LoadData routines. The libraries they use are FileExport.dll and FileImport.dll, but in theory they can load any library supporting the Abstract Class implementation (found in the mentioned libraries). The libraries have been coded for specific file layouts based upon need. For example….to load position data from Phase3 the following batch job command is executed:
  + **C:\Trading\Programs\Batch\bin\fileimporter.exe /file \\mail\sungardarchive\[M\_d\_yyyy]\'invpos' /td1 /dbserver PROP-RN\SQLEXPRESS /dbcatalog Clearing /dbtable Positions\_Phase3 /wipeb4import false /lib FileImport.dll /class Utility.IO.Imports.AFCO.Phase3\_PositionImport /filewatch true /mail rn@albertfried.com /filewatchinterval 60000**
  + The /class and /lib parameters denote the class and respective library the class is located in. other parameters specify file location, database info, etc…
* **MoveFiles** – this was just a short utility to move a directory of files easily to an archive utility. Currently only used to copy the moc fix log data.

***Imports and Exports***

The imports and exports are scheduled and controlled through the Windows 7 Task Scheduler. Here is a current snapshot for the schedules for all things…note that not all are batch jobs…



(save the picture and edit if you want to see a bigger size image)

Obviously there is a dependency chain within some batch jobs. For example, positions cannot be exported to tradespeed until they are loaded to the database…therefore **Export Phase3 Positions To Tradespeed** should not run until after **Import Phase3 Positions** runs. Unfortunately, the jobs are not currently coded to know if a prior job completed successfully or not. Another dependency example is the ETF File Import….that depends on files being downloaded via FTP to the c:\temp\etfs directory via the **Download ETF Files from DTCC** job.

The names in the Task Scheduler are fairly self describing. Drilling down into any of them will get you the .cmd file associated with each, and from there you can glean a bit more information.

***FTP Routines***

The ftp downloads are fairly straight forward in that the **copyftp** executable can handle generic ftp downloads from places such as tradespeed, or a web based file download, and can even handle SFTP in the case of the DTCC requirements. There are two FTP routines that have slightly different requirements, with one requiring maintenance quarterly.

* ETF ftp’s from DTCC – The requirement by the DTCC is that the password be changed every 90 days. The only way to do that currently is via a command line ftp program called moveit….the description on how to do it is located in the notes of my outlook. To change the password, you will need to open a command window and:
  + Cd “c:\program files (x86)\moveit”
  + ftps -e:on -a -z -natpasv fdmovers.dtcc.com 9021
  + to change the password enter usedid tcp0284s hit enter, then enter the currentpassword/newpwd/newpwd. So currently blue90/blue91/blue91.
  + Update the batch job md file in “C:\trading\Programs\Batch\Imports\copyftp\_etf files.cmd” for both entries to reflect the new password and update the outlook not with the new password as well.
* The other non standard ftp is for the RBH download. It cannot be done programmatically because of the nature of how the OCC encypts the connection…it uses their program called ftpplus located in c:\ftpplus. The directory and subdirectories need to remain the same or it will break and no longer run.

**After Trading Batch Jobs**

The after trading hours batch jobs are more of the same in terms of file importing and exporting and are located in the same directory as the other batch routines. The only noted process is the Export\_Trades\_ToBlotter.cmd which will copy the trades located in the database in tables: **BrokerTrades, ManualTrades and Executions** to the **ConsolidatedTradeBlotter** table. The code actually calls into a view Trades\_Executed\_Today\_Summary\_NY81 to gather its data. The view itself ties those 3 tables together for processing.

**Real-time processes**

* DropCopy servers

The servers are quickfix engines wrapped in code specific to the current database structures. The executable is located in **C:\Trading\Programs\Production\dropcopies\DropCopyServer.exe** and uses a command line parameter to specify the settings file to use (which are all located in the same directory. For example…to connect to tradespeed, the following command line is used (within a batch file in the same directory called dc\_tradespeed.cmd):

**cd c:\trading\programs\Production\dropcopies**

**DropCopyServer.exe -s tradespeed\_prod.config**

**QuickFix** is an open source fix engine that handles the cunnication layer and session layer aspects of fix. The messages are then registered for and handled by other code to update data to either the Executions table or the OrdersAndCancels table.

The issues that arise from this are usually of the failed to connect variety. See the troubleshooting section to understand how to fix that. Another issue that may arise over time is if the vendor we are connecting to decides to change the fix messaging in such a way that it breaks the integrity of how data is entered into the database….this will, unfortunately, require coding fixes.

* P3 TradeProcessingEngine

The Phase3 TradeProcessingEngine uses fix to enter trades real-time to Sungards Phase3 System. This engine uses quickfix similar to drop copies, and then uses a couple of libraries written around an interface class to load data from the database. These queries occur every 3-5 seconds and only query for todays trades but with a **tradeid** greater then the last know tradeid it found on the prior query.

The executable is located in **C:\Trading\Programs\Production\TradePostEngines\P3** and is launched via the **p3tpe.cmd** batch command file. It relies on some translation and other settings located in the same directory in file **P3PostEngineConfig.xml**. Mostly account and exchange code translations are used from the file, with an occasional ContraBroker and Equity code translation based upon the connection name.

The program when it starts will load its prior known data for the day from the cache table (see setting entries **CacheSendTable and CacheReceiveTable**)**.** It will then load the scan libraries responsible for querying the various trade tables directly (BrokerTrades and Executions). These scan libraries will query the database every 3-5 seconds for any new trades.

While the main engine is a command line program, a form is associated to show the state of the engine and the responses received from Phase3. **Unfortunately** given the nature of the task scheduler, the program usually runs in the background and hence no form will be visible. If you wish to access the command line utility or the form, you will need to kill the P3TradePostEngine.exe in the task manager, and then launch it manually via **C:\Trading\Programs\Production\TradePostEngines\P3\p3tpe.cmd**

There should be no harm in doing this.

Hitting **?** on the command line will bring up a couple of simple commands.

* MH TradePostEngine

Similar to the P3 TradePostEngine, the MH TradePostEngine relies on database scans to find new trades. It also uses a settings file as well for certain translations needed. The main difference is that there is no fix engine involved, and it instead writes to a database **PROPTRADING-BPM\SQLEXPRESS\Databases\microhedge\Tables\trades**.

The database scans for new trades occur every 3-5 seconds here as well.

* MarketDataRequestService

This process was developed against the Bloomberg 3.x API and utilizes multicast messaging to handle requests usually from the Excel Data Cache library, but it can be from anywhere that conforms to the proper serialized request class. The executable is located at [**\\proptrading-bpm\c$\Trading\automation\mdrs**](file:///\\proptrading-bpm\c$\Trading\automation\mdrs)and the executable MarketDataRequestService.exe

The process also runs the nightly data collection routine which will attempt to get price, dividend, and fundamental data for all symbols located in the database table PROP-**RN\SQLEXPRESS\Databases\SecuritiesMaster\Tables\ ActiveHistoricalSecurities**.

It relies on a view in that database called BloombergHistorical\_LastKnownDate to determine the data range to query bloomberg for….this way the process can catch up if the process has not run for a few days.

A utility located on [\\prop-rn\c$\trading\programs\util](file:///\\prop-rn\c$\trading\programs\util) can be used to request data for a symbol that is for some reason not working in Excel. The program is requestsymbol.exe and can be run from command line or double click. From there type in the symbol name in uppercase….just make sure the MarketDataRequestService.exe is running on the machine with bloomberg.

* Excel Integrated Tools

**Excel Integrated Tools**

Code used to enhance Excel are located in the **C:\Trading\Excel\xladd** and consist of a coordination of several libraries. Excel has an add-in load interface in the registry (HKLM\_Software\Wow64Node\Microsoft\Office\Excel\Addins\ xladd.xladdinmgr)

This is the main load class (xladd.dll class=xladdinmgr). It is responsible for creating menus, loading other addins and libraries and initializing the datacache that Excel uses in conjunction with its worksheets function (those are added differently to Excel via the Addins\Com…).

The datacache is the most used class when it comes to worksheet function analytics. The datacache begins its life by loading historical data from the database (**PROP-RN\SQLExpress\Databases\SecuritiesMaster\Views\ SecurityHistoricalWithDivs\_ActiveList**). It will then massage the data according to the following rules:

* Sort all info by symbol and then by trade date ascending
* Discard any 0 price data
* Calculate percent move between two days incorporating any dividend

From there the data in the cache sits until needed. This is where the COM addins come into play to provide analytic worksheet functions to Excel. The math class provides a c# quantlib wrapper to excel which utilizes the cached data.

When a function that requires historical data is called from a cell, the code will build a callback object to store should data not exist for a given symbol. The code will then call into the cache and request the data with the specified date ranges. If the data is found, it is returned and the data is then massaged into quantlib requirements and the data is returned. Should the data not be in the cache, the callback is stored by the cache, and a request message is sent via multicast to the MDRS (**MarketDatraRequestService**) process to request the missing data. Provided it is running, the MDRS will get the data, insert it to the database, make the requested security active, and send a serialized update back to the requestor, and then a generic update via multicast so that any other cache may update itself. If the cache receives the update before the request times out, it is inserted into the cache, the original requestor callback is notified, and Excel is told to recalculate itself for that particular calling cell.

In theory, very harmonious….in practice subtle issue creep in and it may not work as expected. If data does not propagate then use the utility described in the [**MarketDataRequestService**](#RTPMDRS) section of this document.

***Analytic Functions***

The Quantlib wrapped function that use the datacache have parameters that cannot be described so that Excel can show them in the formula help. Below is the description header from each function:

<System.ComponentModel.Description("Stock Correlations. ReturnValue(0,1,2)=Correlation,Used StartDate, Used EndDate"), System.ComponentModel.Category("Statistics")> \_

Public Function sCorrel(ByVal Symbol1 As String, ByVal Symbol2 As String, ByVal StartDate As Date, ByVal EndDate As Date, Optional ByVal Periodicty As Integer = 1, Optional ByVal ReturnValue As Integer = 0) As Object

<System.ComponentModel.Description("Provides a way to interpolate a curve as return a value on that curve"), System.ComponentModel.Category("Statistics")> \_

Public Function CubicInterp(ByVal xBegin As Microsoft.Office.Interop.Excel.Range, ByVal yBegin As Microsoft.Office.Interop.Excel.Range, <System.ComponentModel.Description("Test")> ByVal xWant As Double, Optional ByVal DebugOn As Boolean = False) As Object

<System.ComponentModel.Description("Stock Covariance. ReturnValue(0,1,2)=Covariance,Used StartDate, Used EndDate"), System.ComponentModel.Category("Statistics")> \_

Public Function sCovar(ByVal Symbol1 As String, ByVal Symbol2 As String, ByVal StartDate As Date, ByVal EndDate As Date, Optional ByVal Periodicty As Integer = 1, Optional ByVal ReturnValue As Integer = 0) As Object

<System.ComponentModel.Description("Stock Beta. ReturnValue(0,1,2)=Beta,Used StartDate, Used EndDate"), System.ComponentModel.Category("Statistics")> \_

Public Function sBeta(ByVal Symbol1 As String, ByVal Symbol2 As String, ByVal StartDate As Date, ByVal EndDate As Date, Optional ByVal Periodicty As Integer = 1, Optional ByVal ReturnValue As Integer = 0) As Object

<System.ComponentModel.Description("Stock Variance. ReturnValue(0,1,2)=Variance,Used StartDate, Used EndDate"), System.ComponentModel.Category("Statistics")> \_

Public Function sVariance(<System.ComponentModel.Description("Test")> ByVal Symbol As String, ByVal StartDate As Date, ByVal EndDate As Date, Optional ByVal Periodicty As Integer = 1, Optional ByVal ReturnValue As Integer = 0) As Object

<System.ComponentModel.Description("Stock Beta. ReturnValue(0,1,2)=Beta,Used StartDate, Used EndDate"), System.ComponentModel.Category("Statistics")> \_

Public Function sStdDv(ByVal Symbol As String, ByVal StartDate As Date, ByVal EndDate As Date, Optional ByVal Periodicty As Integer = 1, Optional ByVal ReturnValue As Integer = 0) As Object

<System.ComponentModel.Description("Stock Ratio. ReturnValue(0,1,2,3,4,5)=Ratio,Used StartDate, Used EndDate,StdDev of ratio,High,Low. UseValue(optional)(0,1,2)=DivAdj%Chng,Close,NonDiv%Chng."), System.ComponentModel.Category("Statistics")> \_

Public Function sRatio(ByVal Symbol1 As String, ByVal Symbol2 As String, ByVal StartDate As Date, ByVal EndDate As Date, Optional ByVal Periodicty As Integer = 1, Optional ByVal ReturnValue As Integer = 0, Optional ByVal UseValue As Integer = 1) As Object

<System.ComponentModel.Description("Index CoVar,Var,AvgVol. IndexData to be blank, or 3 column range of Compnent/Shrs/Last. ReturnValue(0,1,2,3,4,5)=AvgCorr,TotalCovar,TotalVar,Used StartDate, Used EndDate."), System.ComponentModel.Category("Statistics")> \_

Public Function IndexCVAV(ByVal Index As String, ByVal StartDate As Date, ByVal EndDate As Date, Optional ByVal ReturnValue As Integer = 0, Optional ByVal Perodicity As Integer = 0, Optional ByVal DebugOn As Boolean = False, Optional ByVal IndexData As Excel.Range = Nothing) As Object

***Misc***

* Ratio Spreadsheet

The ratio spreadsheet is used to show how correlated pairs are trading with respectsw to the historically computed expected ratio. The sheet is located in:

[**\\prop-rn\c$\trading\shared\ratio spreadsheet.xlsm**](file:///\\prop-rn\c$\trading\shared\ratio%20spreadsheet.xlsm)

**Database**

To view database tables, find SQL Server 2008 folder in you start menu, and select SQL Server Management Studio 2008.

There are two main databases running. One on PROP-RN\SQLEXPRESS and the other on PropTrading-BPM\SQLEXPRESS.

PROP-RN\SQLEXPRESS contains the majority of the historical data, security master, trades and blotter and just about everything else. The db on PropTrading-BPM\SQLEXPRESS contains simply the local microhedge database used for microhedge trades, profiles, accounhts and settings….

* **Used Tables**

Used tables will be reflecting the db on PROP-RN\SQLEXPRESS only.

There are 4 Main Databases on PROP-RN\SQLEXPRESS.

* + *Clearing*
    - **AccountToMicrohedgeAccount\_Map**

Used to translate the Phase3 Account to a muicrohedge account. There is a column for a potential risk symbol map as well, but currently unused.

* + - **Positions\_Phase3**

Used to load all Phase3 client accounts, although the views will only reflect the used account for prop. Things that feed from this…**position mismatch report, pnl report (closing prices)**.

* + - **P3TradePost\_\_RcvCache, P3TradePost\_\_SendCache**

Used by the [P3 TradePostEngine](#RTPP3) to cache its state.

* + - **Views**

Many views feed into each other, and others are used for the batch jobs and other things. Just note that in Clearing, the main view for positions is ***Position\_from\_TradeBlotter*** *and for microhedge loads each day it is* **Microhedge\_Positions\_From\_Blotter.**

* + *SecuritiesMaster*
    - **ActivHistoricalSecurities**

List of securities that will be used in the daily bloomberg scrape, and in the Excel DataCache loading.

* + - **ETFs and ETFComponents**

Imported daily from the NSCC composition file. The associated views (*ETFComponents\_Today and ETFs\_Today*) are used for the Excel ETF drops function.

* + - **Phase3Options**

***Critical table.*** This is used to determine what positions are from Phase3 on a daily basis. Phase3Positions only contain Sungards version of a cusip. This table is the only way to translate between the two.

* + - **Securities**

Contains a list of cusips and securities used by other tables (**Phase3Options on UnderlyingCusip, SecurityDividends,SecurityHistorical,SecurityRefData,SecurityVendorLookup,ActiveHistoricalSecurites**).

Periodically a duplicate comes in when a cusip changes. Much of the code doesn’t concern itself with cusips and so a double record can arise on views/queries that group. If it occurs, the offending record will need to be deleted AFTER the above associated tables have had the cusip changed to the new cusip).

* + - **SecurityDividends**

This table is populated via the [MarketDataRequestService](#RTPMDRS). And is used to calculate daily % moves of a security. Periodically a dividend exdate is not reflected properly and will need to be manually adjusted.

* + - **SecurityHistorical**

Historical prices.

* + - **SecurityVendorLookup**

Used to translate from the Security Table symbol to a marketdata vendor. This is used by the [MarketDataRequestService](#RTPMDRS) to request a symbol’s data.

* + - **ISOCountry, ISOCurrency, ISOMIC**

Not used. Was part of a revamped project to create a better financial instruments OO library in .NET. Data is populated based on the ISO Codes for Country, Currency and Market Center codes. So has future value potential.

* + *Trading*
    - **Algo\_IMB**

Used by the MOC Also to load symbols.

* + - **BrokerTrades**

Manual Entry of broker trades. Will feed real time into Phase3 for clearing via the [P3 TradePostEngine](#RTPP3) and into Microhedge via the [Mh TradePostEngine](#RTPMH).

Will also be copied over at end of day into the *ConsolidatedTradeBlotter table*.

* + - **ConsildatedTradeBlotter**

**Critical table.** Used to house all transations for the NY81 account lifetime. Several queries rely on this data to feed start of day positions, run p&l, etc…

* + - **ManualTrades**

Place to enter manual trades, assignments, exercises that will copy at end of day to the ConsolidatedTradeBlotter, but will not flow to MH or Phase3.

* + - **Executions**

**Critical table**. This is where the [DropCopyServer](#RTPDropCopies) trades will feed to, and they WILL feed live into Phase3 and Mh.

* + - **Holidays**

Simple table of exchange closes in the US. This is used by the Excel worksheet function *TDFromDate* and is also used by a custom database function used in many views called *Trading.dbo.GetPrevTradeDate(daysback)*. This is currently populated through 2013. It will need more entries going forward.

* + - **RegSho**

Used by the regsho batch job to build the regsho list.

* + - **OrdersAndCancels**

Used by the DropCopyServer, but on ly to house any order / cancel messages. Removing or modifying this may cause the DropCopyServer process to crash.

* ***MIcrohedge***

On PropTrading-BPM\SQLEXPRESS \Datbases\microhedge\Tables\Trades, this is the main table for discussion. The trades table contains all trades for all positions. The table itself is wiped and loaded fresh each morning by the task scheduler job **Export Positions To Microhedge DB (Brian)**. This job can be run anytime during the day, but it will **ONLY** load position from start of day.

When manually manipulating this table, be aware that the table uses triggers to protect its data. So when you try to delete a row, in the editor it will be deleted, but when you requery the table, the row will still show intact, but a column called ***SoftDel*** will show as true. The microhedge application trades this as a deleted row and will not show it in the trades table within the app.

Conversely, when trades are supposed to be reflected in Microhedge but are not, check this table to see if the ***SoftDel*** *column is set to True*. If so, change it to False for it to show in Microhedge.

**Note about options with different root codes**. The trades table uses a column called **AltSym** to match up options with different root codes (i.e. RYN1). If an option is not reflecting the proper strike, or perhaps not showing at all, it may be because the **AltSym** was reflecting the standard root code and not the actual root code.

* **Microhedge Risk**

<describe>

SELECT [EntryType]

,[Symbol]

,[Account]

,[ScenarioName]

,[ExpirationDate]

,[Theta]

,[Vega]

,[Delta]

,[Gamma]

,[UnderlyingPrice]

,[Theo]

,[pnl]

,[VolMove]

,[PriceMove]

,[RunDate]

FROM [microhedge].[dbo].[tblMicrohedgeScenarioRisk]

WHERE RunDate='9/9/2011' and EntryType Like 'Monthly%' and Delta <> 0 AND VolMove='0' and PriceMove='0%'

**Process Locations**

* *Drop Copies*

*C:\Trading\Programs\Production\dropcopies*

*run any of the* ***.cmd*** *files for the appropriate vendor.*

*Will cause a crash if the process is running already*

* *TradePostEngines*

*Microhedge - C:\Trading\Programs\Production\TradePostEngines\mhtpe.cmd*

*Phase 3 - C:\Trading\Programs\Production\TradePostEngines\P3\p3tpe.cmd*

*Either process can be killed and relaunched any time. Multiple processes will cause primary key violations in the database.*

* *MarketDataRequestService*

*C:\trading\automation\mdrs\marketdatarequestservice.exe (on* ***Proptrading-BPM****)*

*Single instance should be running.*

* *Batch Jobs*

*C:\Trading\Programs\Batch and then subdirectories \bin \imports \exports \misc*

*All run from cmd scripts with the exception of* ***fifo\_pnl****(bin directory)*

* *MOC Algo*

*Server - C:\Trading\Programs\Production\orderrouter\_moc\saOrderRouter.exe PROD:*

*But better run from the desktop icon* ***1. moc algo server***

*Client - C:\Trading\Programs\Production\Client\saOrdClient.exe PROD:*

*But better run from the desktop icon* ***2. moc client***

*Market Data - C:\Trading\Shared\installs\Rick\older\moc\activtest.exe*

*But better run from the desktop icon* ***3. moc marketdata***

*One executable per process or it will crash. Check fix connectivity* ***(AFFIX)*** *on the client to ensure green checkmark.*

* *FIFO\_PNL*

*This is an executable written to specifically generate the lifetime pnl of all trades for NY81 account. The executable is located in C:\Trading\Programs\Batch\Bin\fifo\_pnl.exe on PROP-RN.*

*It reads the* ***ConsildatedTradeBlotter*** *table directly and buckets trades into a tradedate, firt in / first out basis to generate a* ***Realized*** *and* ***Unrealized pnl****. The* ***Unrealized*** *pnl is calculated using whatever trades are left over, calculating an average price, and then looking up the current closing price from* ***Clearing\Positions\_Today\_81Account*** *view.*

*All the data is written to a date based csv file in the* ***C:\temp*** *directory as* ***positionsMMDDYYYY.csv.*** *Summing up the realized and unrealized columns and then subtracting from the same file for the first of the month will get you MTD pnl.*

**Expiration Cycle**

On PROP-RN, located in C:\Trading\Exercise Sheets is a spreadsheet called **Expiration Sheet.xlsm**

This spreadsheet can handle loading in the next known set of expiring options, and then at the end of day, expire those options to the **ConsolidatedTradeBlotter** by clicking the ‘Expire Options’ button.

You will first need to check the days activity for any trades in any of the expiring options, as they do not get loaded into the spreadsheet at this time. Do this by going to the database view **PROP-RN\SQLEXPRESS\Databases\Trades\Views\** **Trades\_Executed\_Today\_Summary\_NY81**. Right click and then click on ‘Select Top 1000 Rows’. You can then change the query to filter for todays maturity date and apply any quantities to the expiration sheet.

It will run into issue in the following scenarios:

* An option series expires to cash (example: HK). The underlying marketdata will be UD and will not allow the exercise formula to calc. Manually overwrite the price to the cash price…

**NOTE: The macros use the market data and exercise formula from the first row and copies down….make sure it is not the first row when you save again**.

* Also in the case above, the sheet will not convert the stock to cash…that will have to be done in the database. Use the manual trades table for this. A single entry for the stock position should suffice.
* Also in the above case, if all options expire on that same expiration date for a given series…this sheet won’t pull in those other options. So for HK…all Nov, Dec, Jan options changed to expire in Sep. This sheet will only pull in Sep. Use the database view **PROP-RN\SQLEXPRESS\Databases\Trades\Views\TradeBlotter\_All\_Final** right click…SELECT TOP 1000. Add the following to the bottom of the query: **WHERE Symbol=’<sym>’ and NOT ExpirationDate is NULL**. Execute the query and copy the data to a blank sheet and massage it into the spreadsheet. Then the normal expire options button should work.

**Misc**

* *MOC Algo*

The MOC Algo locations are as shown [here](#ProcessLocations)

The Algo Server uses the **PROP-RN\SQLEXPRESS\Databases\Trades\Tables\Algo\_IMB** to load its initial data and parameter set.

The process is broken into 3 parts. Server, Client, and MarketData.

* + **Server**

The server when launched, will read the settings file in the executable directory called **AlgoDefinitions.xml** todetermine where to start loading data from. That will in turn point it to the table mentioned above.

The server will then initiate a FIX connection request to the provider (tradespeed), and start loading symbols and parameters. Lastly it will initialize communication to listen for market data and client changes and request. These occur using multicast.

The server operates the symbol handlers on a single looping thread. When the time for the trades to be generated, it will send out MOC orders for each and await an ACK from the exchange. Provided the MOC is not rejected, it will then initiate the closing side of the trade for the same quantity. If the MOC order is rejected for any reason, the closing orders will not be sent and the symbol is basically a non event.

If the fix connection is not running, or the client does not **Enable** symbols, or the market data is not running, then no orders will go out.

* + **Client**

The Client is a generic container for a few older algos. When you start it, first double click the child window so that it fills the screen in the container. Secondly, check the status at the very bottom for **AFFIX** if it is not green, then the fix engine is not connected. If no connection, then no orders will go out and the algo will not run.

After checking, go to the Algos Menu and select IMBALANCEALGO. If the server is running and symbols are in the database, then it will populate 2 main grids with all the symbols (takes a few seconds).

The imbalance sizes will populate in the right hand grid when the market data is running, and you refresh the grid data (see next paragraph). Right clicking on a row will bring up a drop-down menu. From there you can enable all symbols in one shot. You will have to right click again and click ‘Send all’ in order for your changes to be sent to the server.

Because of some glitch in the 64-bit os versus the multicast code, wildcard subscriptions don’t work, so you will need to refresh the grid manually. To do this, go back to the main container window entitled : Fajiin Container, and click the File Menu, and then reconnect. This will reload the server parameters back to the client. Do this periodically to get the latest updates for the Imbalance sizes.

* + **MarketData**

The marketdata program is a tool written to pull imbalance size and side from NYSE’s RTOB, and converts it into a multicast message and sends it. The server will pick that up and copy the data out and use it for its own logic when the proper time (and other) triggers are met.

* *Docs and etc….*

Most of anything I have is located in c:\trading (and then a related subfolder) or c:\dev\filespecs

After that, any documents are located on [\\mail\scandrop\rn](file:///\\mail\scandrop\rn)

* *Code*

All code is located in c:\dev and then a related subfolder.

**Troubleshooting**

Below are some common occurrences and how to resolve them. Obviously you can give me a call and I can walk you through.

* *Non standard options entry / issues*

Because the BrokerTrades table was not configured for different rootcodes, manual intervention will be needed in order to reflect the proper position. The concept here is to allow the normal nightly process to run and copy all trades to the **ConsilidatedTradeBlotter**, and then the next morning, to manually adjust the rootcode column of the entry for the prior days trade date.

The next morning, the microhedge trades table will not be reflected properly either…[click here](#DBMH) to see how to fix.

* *Position mismatch issues*

The batch process that generates this report works by executing two main queries:

SELECT \* FROM Positions\_Today and

SELECT \* FROM Position\_from\_TradeBlotter

It then stores results to a hash table and attempts to reconcile the 2, spitting out items that don’t into a report which is emailed. The report shows which was not matched on the blotter side, and which was not matched from the clearing side.

The issues that arise usually far into one of the following categories:

* + **Clearing position is off from the blotter position (would show two entries for the same symbol…one for blotter and one for clearing).**

Usually indicates that a trade was not manually booked properly. Could also mean that a trade was rejected from Phase 3 from the P3 TradePostEngine.

Check with Dawn or Nelson and have them correct.

It is possible that if it is a new position, an entry may not show for clearing side at all…if so it is either because of this above, or a duplicate symbol issue in the database (see below)

* + **Clearing shows a cusip beginning with UK (should be a corresponding entry on the clearing side for the same quantity).**

This is caused by a missing option in the PROP-RN\SQLEXPRESS\Databases\SecuritiesMaster\Tables\Phase3Options table.

Query the table for similar option, copy and paste, but DON’t hit enter or it will confiuse the db.

Then go to Internet Explorer (or whatever) and to sungard login: <http://170.132.238.2/firm27/>

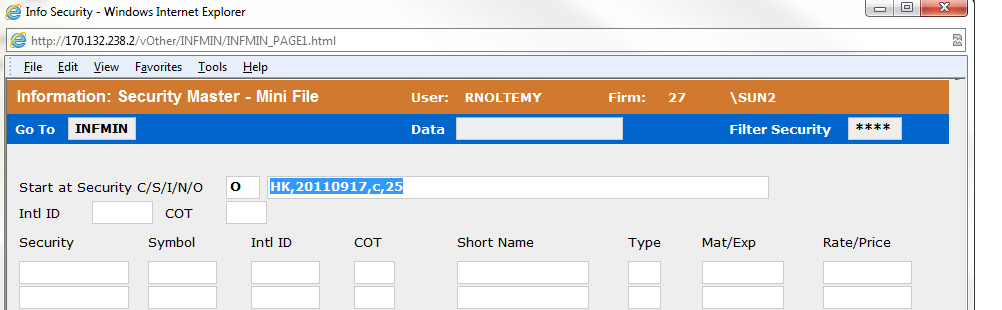
Log in with **rnoltemy** / pwd =

When the next screen pops up, type **infmin** into the the top left text box and hit enter

When that new screen pops up, enter **O** in the C/S/I/N/O box, and in the next box type the symbol info in the following format:

**Symbol,yyyymmdd,callput,strike** and hit enter

This will populate the table below. Copy the cusip, paste it into the cusip field in the database record, and change the corresponding description, expiration data, strike fields and then finally hit ENTER to commit the db record.



* + **Clearing shows a position that does not match the trade blotter side, or trade blotter side does not exist**

Usually an indication of a missed assignment or some other corporate event that caused the position to change without you knowing. Check the trade and enter into the ManualTrades table and it should resolve overnight.

* + **Blotter shows a position with no corresponding clearing side**

This is either from the first issue described above, or it is possible that a security or [phase3option entry is duplicated](#P3OptionDupe). It happens from time to time.

To check if a security is duplicated:

Right click on PROP-RN\SQLEXPRESS\Database\SecuritiesMaster\Tables\Securites, and select Edit Top 200 Rows.

Click the grid toolbar button to bring up the filter criteria pane and enter the underlying symbol and click execute.

If two entries show up, it is usually a corp action cusip change. This is where it gets fun. Because of the foreign key constraints, you will need to change the bad cusip to the good cusip in the following tables:

ActiveHistoricalSecurities

SecurityDividends

SecurityHistorical

SecurityRefData

SecurityVendorLookup

Continued

Use the following query template in a ‘New Query window’ (button top left)

Multiple lines allowed

Using SecuritiesMaster

Update <table> set Cusip=’<newcusip>’ WHERE cusip=’<oldcusip>’

Do that for each table listed above.

Then delete the entry in the securities table with the offending cusip.

If a Phase3Option is duplicated:

PROP-RN\SQLEXPRESS\Database\SecuritiesMaster\Tables\Phase3Options and click Edit Top 200 Rows

Click the grid toolbar button to bring up the filter criteria pane.

Enter symbol, expiration and strike and execute.

Check the entries…it is possible that:

* + - Changed all option expiration dates to the date of a cash offering for a stock

You’ll have to update each of those back to the original expiration date to match up.

* + - Changed cusips for a given series, but forgot to flag the other cusips as deleted

You’ll have to figure out which cusip is good or bad using sungard (see above) and then rename the symbol or delete the offending records. Keep in mind this may have impacted the entire option series….

* + - Did not name the rootcode properly (RYN1 vs RYN)…it happens.

Change the appropriate cusip to the appropriate rootcode (symbol field is the rootcode in this table).

* *Spreadsheet Symbol not updating*

This process is documented [here](#ExcelTools). Just make sure that the MarketDataRequestService.exe is running on Proptrading-BPM (located in c:\trading\automation\mdrs folder).

* *Drop Copy not working*

Drop Copies process is documented [here](#RTPDropCopies).

Currently drop copy servers are running for the following vendors:

Balista

Tradespeed (now Instinet)

Knight Direct

WEX

ISE

The way fix works, the connection is defined by a connection ip address and port, and also a SenderCompId and TargetCompId (synonymous with a username of sorts). All transactions are logged with those 2 fields as well as all the other pertinent trading fields. For troubleshooting purposes you will need the following information is you ever have to call a vendor about a problem with connecting (all of the following are from the individual ***.config*** files located in [\\PROP-RN\c$\Trading\Programs\Production\dropcopies\](file:///\\PROP-RN\c$\Trading\Programs\Production\dropcopies\) folder):

[Balista]

[SESSION]

ServerType=BALLISTA PROD -DROPCOPY

BeginString=FIX.4.2

SenderCompID=AF01

SenderSubID=AF01

TargetCompID=BALZ

TargetSubID=BALZ

SocketConnectPort=17029

SocketConnectHost=209.148.38.223

[Tradespeed]

[SESSION]

ServerType=TRADESPEED -DROPCOPY

BeginString=FIX.4.2

SenderCompID=AFC03

SenderSubID=AFC03

TargetCompID=TORC03

TargetSubID=TORC03

SocketConnectPort=28670

SocketConnectHost=170.16.184.155

[Knight Direct]

[SESSION]

ServerType=Knight Production

BeginString=FIX.4.2

SenderCompID=AFCODC

SenderSubID=AFCODC

TargetCompID=DTTXDC

TargetSubID=DTTXDC

SocketConnectPort=6482

SocketConnectHost=205.167.86.220

[WEX]

[SESSION]

ServerType=Wex PROD -DROPCOPY

BeginString=FIX.4.2

SenderCompID=ALBERTFRIED

SenderSubID=ALBERTFRIED

TargetCompID=WEXXDROP

TargetSubID=WEXXDROP

SocketConnectPort=8049

SocketConnectHost=66.150.108.123

[ISE]

[SESSION]

ServerType=ISE NEW PROD -DROPCOPY

BeginString=FIX.4.2

SenderCompID=AFCDC

SenderSubID=AFCDC

TargetCompID=ISE

TargetSubID=ISE

SocketConnectPort=21502

SocketConnectHost=207.231.197.38

The contact numbers for each of the vedors to troubleshoot are:

* + Ballista - 866-240-9344
  + Tradespeed – 866 323 6158
  + Knight Direct – 888-302-9197
  + WEX – 312-884-3066
  + ISE - 212.897.0244 - Option #2

***How to determine if the connection is up or not***

The dropcopy servers run in the background and hence you will not be able to see any messages on the command console. So you will have to check the logs for each. The logs are generated from quickfix (opensource) automatically and are named using the **SenderCompId-TargetCompId.event.log** and these are located in the [\\prop-rn\c$\trading\programs\production\dropcopies\initiator\_log](file:///\\prop-rn\c$\trading\programs\production\dropcopies\initiator_log) folder). Open the **.event.log** for the appropriate connection and scroll to the bottom. Usually, if connected you would see a couple of messages similar to:

20110915-12:00:05 : Initiated logon request

20110915-12:00:06 : Received logon response

When there is a connection issue, you will see messages that look like this instead

20110914-22:58:32 : Connecting to 170.16.184.155 on port 28670

20110914-22:58:33 : Disconnecting

20110914-22:59:02 : Connecting to 170.16.184.155 on port 28670

20110914-22:59:03 : Disconnecting

20110914-22:59:32 : Connecting to 170.16.184.155 on port 28670

20110914-22:59:33 : Disconnecting

Quickfix open source library will constantly attempt to connect whenever a connection fails.

If a connection has failed, you will need to contact the appropriate vendor, give them the connection information and tell them the fix server for drop copies is unable to connect. In most cases, their server which is hosting has a hung port…happens. Once they restart, the dropcopy servers will automatically reconnect, and download any missing trades (as denoted by a sequence numbering system within fix), so all should be good.

In the case of the **ISE** and **Knight** there is an added layer of a site-site VPN tunnel that has to be established between us and them. While not likely, it is possible that the tunnel has hung. In this case you would have to contact jensyn to handle looking into it….:

Contact [orourkep@jensyn.com](mailto:orourkep@jensyn.com)

* *Entering wrong data into the broker trades table*

Because the Microhedge and Phase3 TradePostEngines pull out trade information within a 1-5 second window from DropCopies and BrokerTrades….and committed trade (one where you’ve hit enter on the database row, or clicked off of the row) should be considered committed to both P3 and MH. The table is setup with constraints to prevent lower case entering, as well as invaluid c/p and buy/sell fields as well, but the Symbol, Price and Quantity cannot be constrained and there may be issues. In order to correct for these, there are two main ways to do it:

* + **Have the back office group cancel and correct the trade**

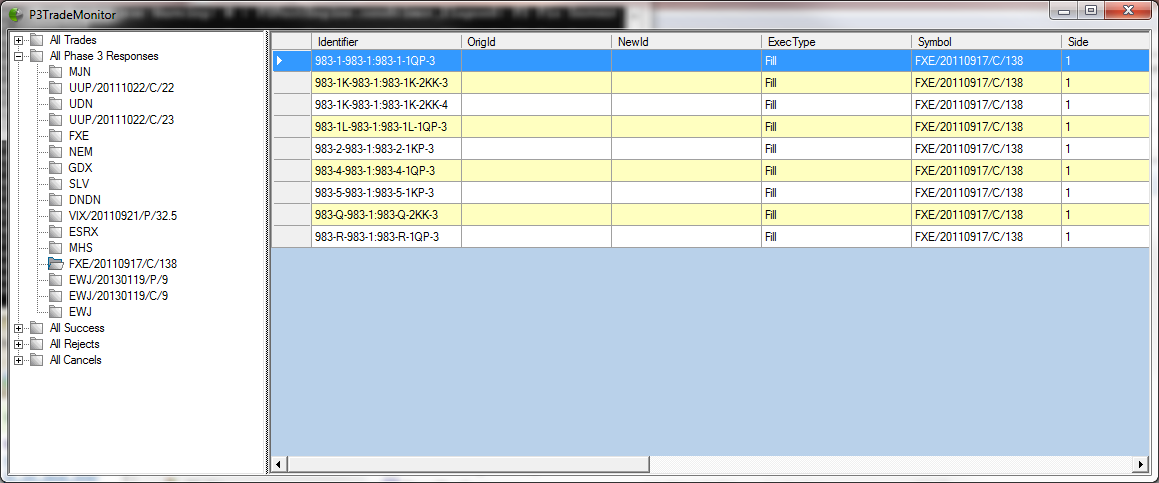
Having Dawn or Nelson correct the offending trade is easiest. Notify them of the error trade symbol, side, qty, price and then give them the current trade details to book.

You will have to correct the entry in Broker Trades table, and manually correct the microhedge tradelog as well.

* + **Correct the trade yourself**

You will need the P3 TradeProcessingEngine GUI for this, which means you may have to kill the background process and then relaunch…see [here](#RTPP3)

You will need to select the ‘All Phase 3 Responses’ folder in the treeview to the left. Hit the + sign on that and then look for the bad trade in the folder list and click on that folder…see picture below:



Now find the offending trade in the list of trades in the grid to the right, click it to select, and then right click to bring up the menu. Select **Cancel Trade**. A dialog box with the trade information will pop-upo asking to confirm. Click ok if the trade details are what you are looking to cancel, and this will send a cancel to Phase3. Which will also create another row entry in the grid you are looking at.

Once that is done, delete the offending trade from the Broker Trades table, and re-enter the correct trade as a new entry.

Now…this will also create another entry into the Microhedge trades table (**PropTrading-BPM\SQLEXPRESS\Databases\microhedge\tables\trades**). You can either manually adjust in the MH application, or edit the trades table and delete the offending trade in there….see [here](#DBMH).

* *Entering wrong data into the manual trades table*

Because the manual trades are processed only once at the end of day (arounde 7:30 pm), you can correct the entry in there without any issue.

* *Historical Data cache not handling split information properly*

The datacache process description is [here](#ExcelTools).

Once way to check to see if the data for a symbol is incorrect after a dividend or split, use the **Historical** add-in function in Excel. It uses a symbol, start date and end date and then drops all the data into the rows below the cell with the function =Historical(A1,startdate,enddate).

Look at the data around a suspected date and see if it was handled properly. If not, it is most likely an issue with how the dividend table got the data.

Edit the database table

**PROP-RN\SQLEXPRESS\Database\SecuritiesMaster\Tables\SecurityDividends**.

Filter by symbol, and sort by ExDate descending.

Sometimes the bloomberg API will put what we think should be the ExDate as the PayableDate.

Usually, you can just adjust these fields manually and then relaunch a new instance of Excel.

Sometimes because the bloomberg process hasn’t run for a few days, and because bloomberg will adjust historical data to reflect the new pricing, data for given days before the split may be reflected improperly and will have to be adjusted manually in the **SecurityHistorical** table.