FactSet OnDemand UserGuide for MATLAB and R

Version 3.3 A

Table of Contents

Chapter 1. Introduction

1.1 Overview

FactSet supports the major tasks frequently undertaken in a Statistical Package data mining project, such as data acquisition, data preparation, modeling, and model execution with reporting/graphing. The integration of data from FactSet into the Third party program allows users to interact with Statistical Package directly to retrieve company, portfolio, index, market aggregates, ownership and economic data from FactSet.

The integration into a statistical package provides access to FactSet databases through a connection to the FactSet OnDemand servers via an https request over the Internet.

R and MATLAB users can access functions from FactSet called factlets to retrieve a variety of data sets. Factlets are functions that encapsulate business logic and data collection procedures. A factlet can be a simple data request or can invoke complex application logic. The technology is capable of cross referencing and processing time-series for a high amount of data, which can be returned in a variety of tagged or delimited formats.

1.2 FactSet Data

FactSet integrates over 200 databases to analyze markets, companies and industries. For a comprehensive list of databases available on FactSet go to: www.FactSet.com

1.2.1 Company Level Data

Highlighted below is a selection of the FactSet owned databases:

- FactSet Fundamentals
  - Access current, comprehensive, and comparative information on securities worldwide with FactSet Fundamentals. The comprehensive coverage available on FactSet Fundamentals includes more than 73,000 companies, 20 years of historical data, and up to 2,000 data elements on each company record. Comprised of annual and interim/quarterly data, detailed historical financial statement content, per share data, calculated ratios, pricing, and textual information, FactSet Fundamentals provides you with the information you need for a global investment perspective.
- FactSet Estimates
  - FactSet Estimates combines a great breadth of data with a level of transparency that ensures you are using the timeliest and highest quality data available in the industry. Access comprehensive consensus-level estimates and statistics with daily history.
- FactSet Ownership
  - The FactSet Ownership database collects global equity ownership data for institutions, mutual fund portfolios, and insiders/stakeholders. FactSet provides both summary and detailed ownership data that can be viewed by security or by holder (institution, mutual fund, insider/stakeholder, as well as beneficial owner in the case of UK domiciled securities).

1.2.2 Benchmark/Index data

FactSet clients have access to Equity and Fixed Income Benchmarks, which include Dow Jones, FTSE, MSCI, Russell, S&P, Barclays, and BofA Merrill Lynch, among a number of others.

FactSet Market Aggregates (FMA), combines data from FactSet Fundamentals, Estimates and Prices to calculate ratios and per share values on an aggregate level. FMA provides access to over 50 metrics for more than 3,500 commercial and exchange indices.

1.2.2 Economic Data

FactSet Economics database is updated intra-day and provides comprehensive global coverage. Database includes details such as economic information, exchange rates, commodities, and interest rates. The database provides:

- Coverage of more than 200 countries.
- Includes over 1.9 million data series.
- In addition to data sourced directly from statistical agencies and organizations, specialty databases include Eurostat, ICIS, ICAP, International Monetary Fund (IMF) and the OECD.

1.3 Data Extraction from FactSet

As outlined above, FactSet integrates a range of datasets. The data is stored on FactSet in such a way as to make powerful cross-sectional and time-series analysis possible, the different approaches to download data from the FactSet mainframes are outlined in Chapter 3.

1.3.1 Technology

The FactSet OnDemand integration generates a URL that creates an https request. The data request will be transmitted over the internet to FactSet OnDemand Servers. FactSet OnDemand uses HTTP basic authentication over Secure Sockets Layer (SSL). The OnDemand servers handle authentication of the user and the permissioning of data sets. The OnDemand integration is designed to provide simple access to FactSet data in reasonably sized blocks through a web service.

1.3.2 Defining the Result

Requesting a large amount of data requires defining what is needed, breaking it down into smaller bundles, retrieving them, and reassembling it on the receiving side. An application has to focus first on defining what a user wants to receive. Only by targeting a very well defined result, can the application select what it needs from the database, break it down, and move it efficiently

1.3.3 The “Ideal” Data Structure

Given a client’s interest in large amounts of data and the limitation of any package in terms of memory to store an unlimited amount of data, FactSet defines that an “ideal” data structure would have a limit of 1 million data points. A large request that is beyond the 1 million data points limit would need to be broken down into smaller pieces in order to move it and then store it permanently.

There are numerous ways to do build a permanent data structure. One large streaming download may not be the answer, because there are limitations at both FactSet and the end usernd in between delivery system. If there is a relatively inflexible query, streaming may work, but large flexible queries require time to build and may need to be broken up for efficiency. With a steady stream of data, the receiving application would have to stop and write it permanently to the database once in a while. Also, with a large transfer, there is the risk of failure.

Sample scripts are provided in the Appendix with examples of how to break down a request, with comments added to explain each line of the script.

Chapter 2. FactSet OnDemand plugin installer

The FactSet OnDemand integration allows users to retrieve FactSet data within Third Party software.

System Requirements:

1. Minimum MATLAB version 2010b or R version 3.0 is required
2. The .NET framework 4.0 is the minimum framework requirement.
3. The MATLAB DataFeed Toolbox is a requirement for extracting FactSet data in MATLAB
4. Pre-release versions of MATLAB and R are not officially supported.

Note: The FactSet MATLAB/R Plugin is retired 18 months after release. The Plugin will continue to work after 18 months, but from 15 months and on users are prompted to upgrade.

The installer makes several changes to a user's MATLAB/R install:

1. Creates a directory under the MATLAB/R root directory called 'FactSet3' where the required files for the plugin are stored.
2. Adds instructions to or creates the startup.m/Rprofile.site files.
3. Adds required information to the path to support the plugin.
4. If .NET 4.0 is not installed it will be installed
5. In R the library rClr is installed

Additionally, in terms of the firewall specifications the below must be allowed

1. Ensure to whitelist https://*.factset.com on firewalls / URL inspectors / IPS / IDS;
2. Ensure that https:// (TCP 443) is allowed to FactSet's production destination subnets - 192.234.235.0 (255.255.255.0) and 64.209.89.0 (255.255.255.0)

Note: Plugin version 3.0 can be installed in conjunction with a previous OnDemand plugin on the same Matlab or R version as long as the above requirements are met.

2.1 Accessing the Installer

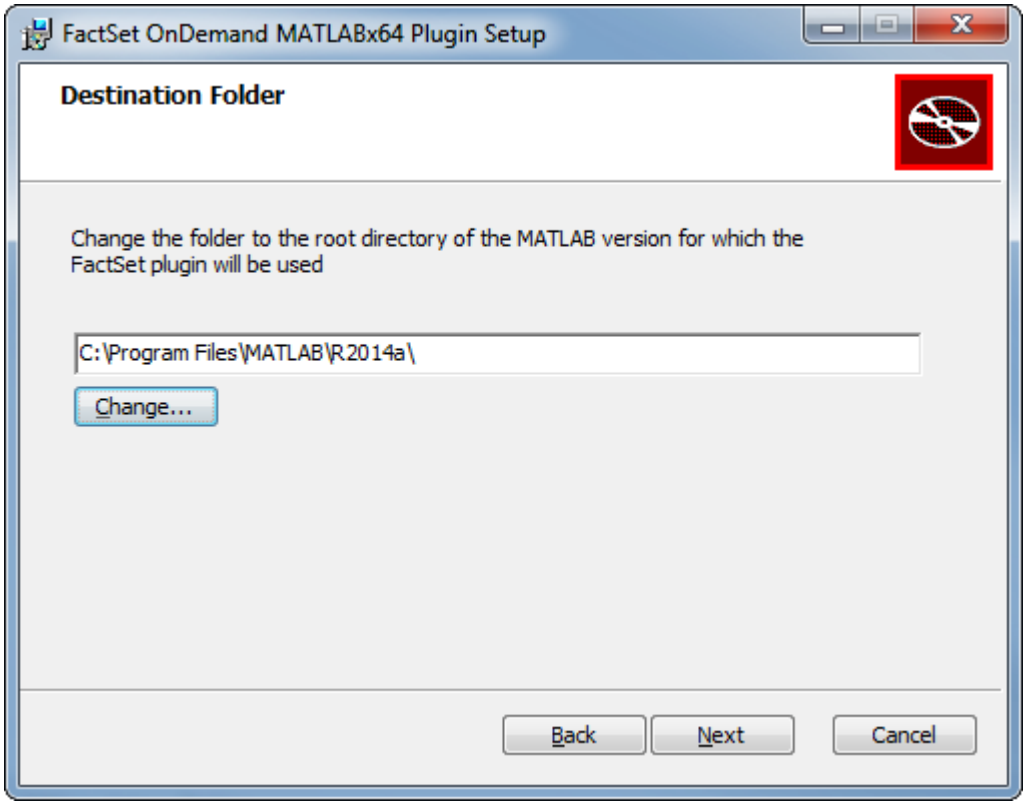
The FactSet OnDemand Installers can be accessed from: https://www.factset.com/download

The plugin is available for 32bit and 64bit architecture, ensure to select the installer that corresponds to the software version the plugin should be used, i.e. if for a 32bit MATLAB version the FactSet plugin installer for MATLAB 32bit should be used (regardless of the PC's architecture).

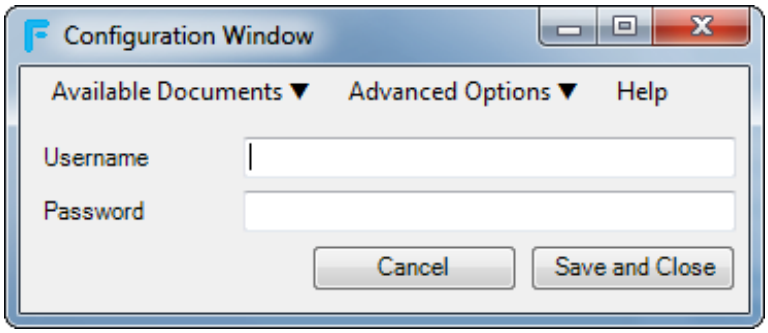
2.2 Running the Installer

The installer comes as an .msi file, running the msi will prompt to select the Destination Folder; here it is required to change to the root directory of the MATLAB or R version being used. For example, the root could be: "C:\Program Files\MATLAB\R2014a". This folder is the location in which the files will be installed.

Note: if the program does not detect a proper root, it will not install.



Once the install is completed a blue F-icon, will appear in the system tray, by double clicking this icon a Configuration Window will open up where the user-specific settings should be added. Every user has a unique username and password (linked to the FactSet terminal serial number), these are entered in the main page of the configuration window. Advanced details such as proxy settings can be accessed in the Advanced Options Configuration window. By default the proxy settings are inherited from Internet Explorer.

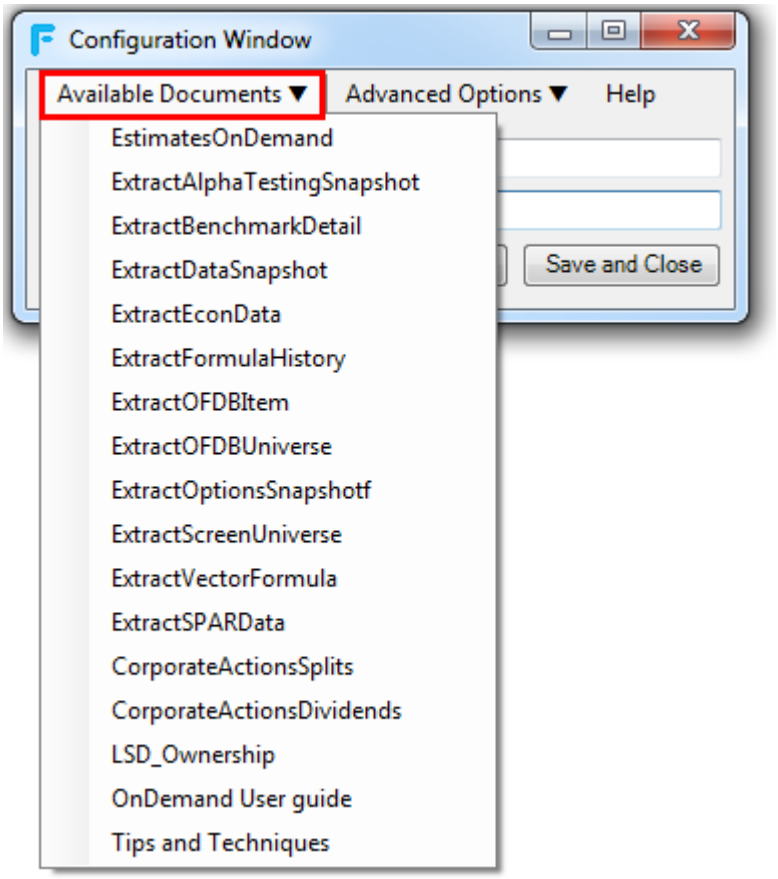


2.3 Help and Support

To uninstall the program or apply repairs, from the Control Panel in Windows, go to **Programs – Uninstall or change a program** and view the entry there.

2.3.1 Documentation

For extensive and up-to-date documentation refer to the Configuration Window – Available Documents. A current Internet connection is required as these documents are stored centrally by FactSet.



2.3.2 Event logger

A built in troubleshooting tool for the plugin is the FactSet Event logger. The event logger captures information that will enable further troubleshooting.

To run the Event logger use the command

F.DumpEventLog()

This will save the EventLog.txt file to the default folder. For MATLAB, the default folder is Documents/MATLAB. For R, the default folder is Documents.

To save to a specified directory, please use the following command:

MATLAB:	**F.DumpEventLogTo('C:\Users\xxxxxx\Desktop\Test.txt')**
R:	**F.DumpEventLogTo('C:\\Users\\dsheldon\\Desktop\\Test2.txt')**

2.3.3 FactSet Support

FactSet troubleshooting questions can be directed to FactSet Support at https://www.factset.com/support-numbers

Chapter 3. Retrieving FactSet data in a Statistical package

3.1 Extract Data from FactSet

FactSet stores all of the available data in proprietary database structures on FactSet servers. This allows FactSet to adjust the way data is stored, so that clients can access data as efficiently as possible. Most datasets available on FactSet are stored in two different ways, so as to facilitate two different data access methods. These two options use the FactSet Query Language (FQL) for time-series requests and the FactSet Screening Language (Screening) for efficiently extract data for a large universe of securities as of a single date. Please see Online Assistant page 10410 for further information on the storage methods.

3.1.1 FactSet Query Language (FQL)

The extraction of a time-series of data from FactSet can be done using the FactSet Query Language (FQL). FQL is a proprietary data retrieval language used to access FactSet data. For more information on FQL, see the FactSet Online Assistant page 1961.

To request a time-series of data; a start date, end date and frequency need to be specified. If a date is not specified, data is returned from the most recent time period. The dates can be either absolute dates or relative dates.

Some advantages of FQL include:

The ability to specify dates for any database using the same formats.
With FQL, date formats are flexible. You can use a number of consistent date formats (defined by FQL) for all databases which makes using and combining data from different databases simple.

The ability to iterate items, formulas, and functions at any frequency.
With FQL, you can iterate items, formulas, and functions at any frequency. For example, you can request a series of weekly price to earnings ratios.

3.1.2 Screening Language

Alternatively, the extraction of data for a list of ids for 1 date, both for equity and fixed income securities is best done using the Screening Language. The FactSet Screening Language is a way to efficiently extract data for a large universe of securities as of a single date.

By default, the Screening Language does not allow iteration and therefore cannot be used to return a time series of data with a single request code. To request data as of a single historical date, an absolute or relative date can be specified.

Overall, FQL syntax should be used to retrieve data for many data items, or to download time series data. Screening syntax should be used to retrieve data for a large universe for a single point in time.

3.2 Dates in FactSet

Dates in FQL and Screening can be expressed as either and Absolute dates such as December 31st 2004 and a Relative date such as latest Fiscal Year end.

3.2.1 Date Format - Absolute Dates

Absolute dates indicate a specific day, month-end, fiscal quarter-end, calendar quarter-end, fiscal year-end, or calendar year-end as depicted in the examples below:

+ A day: MM/DD/YYYY (e.g. 7/11/1999)

Note: *DD/MM/YYYY is not a valid date format.*

- A month-end: MM/YYYY (e.g. 6/1999)
- A fiscal quarter-end: YY/FQ or YYYY/FQ (e.g. 1999/1F, 2000/3F, 2001/2F)
- A calendar quarter-end: YY/CQ or YYYY/CQ (e.g. 1999/1C, 00/3C, 2001/1C)
- A fiscal year-end: YY or YYYY (e.g. 2000, 01, 1999)

3.2.2 Date Format - Relative Dates

Relative dates represent a date relative to the most recently updated period. For example, 0 (zero) represents the most recently updated period and -1 represents the time period prior to the most recently updated.

The zero date is determined by the default time period or the natural frequency of the data being requested. Zero (0) when used with monthly data indicates the most recent month end. Negative one (-1) when used with annual data indicates one fiscal year prior to the most recently updated year.

List of Relative Date Arguments:

D	0D is the most recent trading day, -1D is one trading day prior.
AW	0AW is the most recent trading day, -1AW is the one actual week (7 days) prior to the most recent trading day.
W	0W is the last day of the most recent trading week (usually Friday), -1W is the last trading day of the prior week.
AM	0AM is the most recent trading day, -1AM is the same day, one actual month ago.
M	0M is the last trading day of the most recent month, -1M is the last trading day of the prior month.
AQ	0AQ is the most recent trading day, -1AQ is the same day 3 months prior
Q	0Q is the last trading day of the company's most recent fiscal quarter, -1Q is the last day of the prior fiscal quarter.
CQ	0CQ is the last trading day of the most recent calendar quarter (March, June, September, or December), -1CQ is the last trading day of the prior calendar quarter.
AY	0AY is the most recent trading day, -1AY is one actual year (365 days) prior
Y	0Y is the last trading day of the company's most recent fiscal year, -1Y is the last trading day of the prior fiscal year.
CY	0CY is the last trading day of the most recent calendar year (the last trading day in December), -1CY is the last trading day of the prior calendar year.

3.3 FactSet OnDemand Functions

FactSet has written a number of functions for the FactSet OnDemand plugins that retrieve FactSet data using FQL or Screening syntax by calling stored procedures (factlets) on the FactSet OnDemand servers. Factlets are server-based functions that encapsulate business logic and data collection procedures. A factlet can handle a simple data request or can invoke complex application logic. The data items that are requested, along with an identifier and date, are stored in a structure that is returned.

3.3.1 OnDemand Factlet Requests

The following is a list of the factlets available using FactSet OnDemand MATLAB and R integrations. The description for each factlet also highlights if the factlet should be used with FQL or Screening syntax.

The factlets should be chosen depending on the dataset required. There are generic factlets using either actual Screening or FQL codes as input (to find the correct code please use the FactSet Sidebar look-up dialog) and specialized factlets used for specific datasets.

3.3.2 Standard Factlets

The Standard Factlets below are used for Screening data, Economics data and FQL data. For the exact input syntax, the FactSet Sidebar dialog box can be used.

Factlet	FactSet syntax used by factlet
ExtractDataSnapshot *Function is used for extracting one or more items for a list of ids for 1 date, both for equity or fixed income securities. Should be used to efficiently extract data for a large universe of securities as of a single date.* *The data can also be retrieved using a backtest date to avoid look-ahead bias in the analysis. The backtest functionality is available to clients subscribing to one of FactSet's quantitative applications in the workstation, such as Alpha Testing or Portfolio Simulation.*	Screening
ExtractEconData *Function provides access to a broad array of macroeconomic content, interest rates and yields, country indices and various exchange rate measures from both the FactSet Economics and the Standardized Economic databases.*	FQL
ExtractFormulaHistory *Function is used for extracting one or more items for one security, an index or a list of securities over time.*	FQL

3.3.3 Specialized Factlets

The specialized Factlets are developed for different content sets or specialized data structures. These factlets have been developed to simplify and standardize the data retrieval of more complex data structures.

Factlet	FactSet syntax used by factlet
CorporateActionsDividends *Function is used for extracting stock dividend information.*	FQL
CorporateActionsSplits *Function is used for extracting stock split information.*	FQL
EstimatesOnDemand *Function provides access to FactSet sourced company level estimates data. The data is accessed through the following reports that are available with this function: Actuals, BrokerDetail, BrokerSnapshot, Consensus, Guidance, Surprise, Detailed Recommendations and Consensus Recommendations.*	FQL
ExtractAlphaTestingSnapshot	FQL

Function provides access to data from AlphaTesting model results. Alpha Testing is a tool available in the FactSet workstation used to assess the relationship between one or more variables and subsequent returns over time. A subscription to Alpha Testing in FactSet is necessary to extract this data in the stat packages.	
ExtractBenchmarkDetail *Function is used for extracting multiple data items for a benchmark. Benchmark data can be retrieved using other functions, such as with ExtractFormulaHistory, but the ExtractBenchmarkDetail function allows a user to retrieve a more comprehensive overview of the index constituent data, without additional codes or calculations. In the default output, identifiers are sorted in descending order by weight in the index and each row shows the index id, company id, date, ticker, and weight. Additional items are displayed at the end.*	Screening ***Note**: The Extract-BenchmarkDetail function by default uses Screening codes entered in the items argument of the syntax. If using an FQL code, enter an _ before the FQL items code.*
ExtractOFDBItem *Function provides access to a list of securities and multiple data items for a range of dates uploaded into a single Open FactSet Database (OFDB).*	Screening ***Note:** The Extract-OFDBItem function by default uses Screening. FQL should be used when using ids with spaces or short positions, indicated in the OFDB with an _S.*
ExtractOFDBUniverse *Function provides access to a list of securities belonging to a single Open FactSet Database (OFDB) file as of a single date.*	FQL
ExtractScreenUniverse *Function used for extracting a list of Identifiers stored in a single FactSet screen. In the FactSet workstation, a user can screen for securities based on specified criteria and store the result using FactSet Universal Screening for equity or debt securities.*	Screening
ExtractOptionsSnapshot *Function is used for extracting options data for one or more conditions from the FactSet-Options Derived Values database.*	FQL
ExtractSPARData *Function is used for displaying SPAR data for specified funds from databases that includes S&P, Lipper, Morningstar, Russell, eVestment, Nelson, Rogerscasey, and PSN. A subscription to SPAR in FactSet is necessary to be able to extract this data in statisical packages.*	FQL
ExtractVectorFormula *ExtractVectorFormula function is used for extracting FactSet data that is stored in a vector data format, where the data array does not have a predefined size and is organized by the vector position. A vector can be thought of as a list that has one dimension, a row of data. A vector position allows for a particular element of the array to be accessed.* *ExtractVectorFormula handles non-sequential data with support for matrix or vector output. The*	FQL

nature of the data determines if the output is a matrix or vector, it is not specified in the function to choose which format the data is returned in. This type of data includes corresponding geographic or product segment breakdowns for a company or detailed broker snapshot or history estimates/analyst information.	
LSD_Ownership *FactSet Ownership database collects global equity ownership data for institutions, mutual fund portfolios, and insiders/stake holders. Detailed ownership data can be extracted by company or by holder (institution, mutual fund, and insider/stake). The LSD_Ownership function is used for extracting one or more data items from the FactSet Ownership database for one or multiple securities or holders.*	FQL

3.3.4 Other FactSet Functions

Factlet	FactSet syntax used by factlet
TickHistory *Requires FactSet plugin version 3.1+* *TickHistory is used for extracting real-time trading details for a specific security. The data comes from FactSet's Time and Sales database, which provides history of quotes and trades for a trailing 60 days, or up to 1 year with an additional subscription.*	FQL
Streaming Realtime Exchange Data *Requires an additional subscription and FactSet plugin version 3.0+ for MATLAB and 3.1+ for R/Developer's Toolkit* *The realtime function is used to stream realtime exchange data and will update with each trade.*	FQL
Documents *Requires an additional subscription and FactSet plugin version 3.1+* *The Documents service provides access for the retrieval of news stories, investment research reports, filings, and transcripts. When requested, summary information of the documents will be returned, including an http URL to access the resulting documents.*	FQL
Snapshot *Requires an additional subscription and FactSet plugin version 3.1+* *The Snapshot service provides access to streaming exchange data and allows the "snap" of real-time prices at the user's request. When the command is run, a list of all available exchange data items will be returned by default.*	FQL
F.Cancel() *Requires FactSet plugin version 3.2+* *Using the new F.Cancel() function will now cancel a user's request from the backend, allowing another request to be made.*	

3.3.5 Factlet syntax

For each factlet there are required arguments and optional arguments, these arguments vary between the factlets. Below the factlets are listed with their respective available arguments, for more detailed information about each factlet please see the factlet specific documents available in the Online Assistant.

3.3.5.1 F.ExtractDataSnapshot

The syntax for the F.ExtractDataSnapshot function is:

data = F.ExtractDataSnapshot(ids, items, date, optional arguments)

where,

ids	CellString array with a list of one or multiple security identifiers
items	CellString array with a list of one or more FactSet data items in the Screening language
backtestDate	The backtest date as of which the data is retrieved. If no date is specified, a backtest date will not be set. The date can be entered using a relative date or absolute date.

Optional arguments,

currency	The currency in which the data is to be returned, using a string with the three character ISO code (e.g. 'USD' or 'EUR').
cal	Calendar setting, arguments include: **FIVEDAY:** Displays Monday through Friday, regardless of whether there were trading holidays. **FIVEDAYEOM:** Displays Monday through Friday including a weekend date if it falls on the last day of the month. Where the month-end does not fall on a weekend, the calendar will act just as the standard five-day calendar. **SEVENDAY:** Displays Monday through Sunday. **AAM:** For Exchange code uses the calendar of a specific exchange, represented by the exchange code. If there is no calendar available for a specific exchange, the calendar will default to FIVEDAY.
universe	Screening expression to limit the universe
ison	Ison-codes can be used to limit the universe ISON_MSCI_WORLD(0,1) is written as 'ison','msci_world','isonParams','0,1'
isonParams	The arguments within brackets in the ison-code
OFDB	Universe is the constituents of an OFDB file, default directory is Client, if the OFDB is stored in another location the path must be included
OFDBDate	Specific date for the constituents of the OFDB
universeGroup	Specifies what mode of screening to use. The default screening mode is Equity. For Fund screening and Debt screening the universeGroup argument has to be used with either FUND or DEBT respectively.
decimals	Positionally set according to the items in the selection, ie 'decimals',',,3,4,3'

Example

This example is using the standard Screening syntax to retrieve the quarterly sales value from the FactSet Fundamentals database for IBM using the Screening code FF_SALES (QTR,20110401,RF,EUR). The data is retrieved in currency set to Euro, as of 04/01/2011. The RP default argument in the FactSet Fundamentals database codes reflects that the data is the Latest Preliminary for the Reported Period (alternative arguments could be for example RF, for the Latest Fully Reported Period, among others).

data = F.ExtractDataSnapshot('IBM','FF_SALES(QTR,20110401,RP,EUR)', '20110401')

With the output:

R:

```
  Id      Date ff.sales
1 IBM 2011-04-01 17982.83
```

MATLAB:

```
      Id: 'IBM'
    Date: 734594
ff_sales: 1.7983e+04
```

3.3.5.2 F.ExtractFormulaHistory

The syntax for the F.ExtractFormulaHistory function is:

```
data = F.ExtractFormulaHistory(ids, items, date, optional arguments)
```

where,

ids	CellString array with a list of one or multiple security identifiers
items	CellString array with a list of one or more FactSet data items in the Screening language
dates	Date range and frequency entered using actual or relative dates. A valid FactSet frequency (e.g. 'd'dates Alternate method of entering dates entered in start:end:freq format. (e.g. '20101215:20110115:d')

Optional arguments,

currency	The currency in which the data is to be returned, using a string with the three character ISO code (e.g. 'USD' or 'EUR').
cal	Calendar setting, arguments include: **LOCAL:** Uses the local trading calendar for each security. Local exchange holidays will be skipped **FIVEDAY:** Displays Monday through Friday, regardless of whether there were trading holidays. **FIVEDAYEOM**: Displays Monday through Friday including a weekend date if it falls on the last day of the month. Where the month-end does not fall on a weekend, the calendar will act just as the standard five-day calendar. **SEVENDAY:** Displays Monday through Sunday. **AAM:** For Exchange code uses the calendar of a specific exchange, represented by the exchange code. If there is no calendar available for a specific exchange, the calendar will default to FIVEDAY.
universe	Screening expression to limit the universe
ison	Ison-codes can be used to limit the universe ISON_MSCI_WORLD(0,1) is written as 'ison','msci_world','isonParams','0,1'
isonParams	The arguments within brackets in the ison-code
OFDB	Universe is the constituents of an OFDB file, default directory is Client, if the OFDB is stored in another location the path must be included
OFDBDate	Specific date for the constituents of the OFDB
decimals	Positionally set according to the items in the selection, ie 'decimals',',,3,4,3'
dataType	The optional argument allows users to define a data type for a data item column that is NA for the entire column. This option must be defined for every column/data item requested in the command if it is used at all.
feelback	Setting to control data is not falling forward and display NAs instead of carrying forward values, for those databases that do so (using 'feelback','n').
refresh	This will refresh the connection to FactSet servers to capture the latest database updates. This only needs to be used when a refresh is necessary. It is not recommended to leave this argument in every request made. To use this, the refresh argument should be paired with the value "Y".

Example

In this example extract the last 6 quarters EPS for Exxon Mobile (ticker XOM) using the FQL code FG_EPS.

```
data = F.ExtractFormulaHistory('XOM','FG_EPS(0Q,-5Q,Q)','0Q:-5Q:Q');
```

With the Output:

R:

```
  Id      Date fg.eps
1 xom 2012-12-31   9.47
2 xom 2013-03-31   9.70
3 xom 2013-06-30   9.82
4 xom 2013-09-30   7.96
5 xom 2013-12-31   7.66
6 xom 2014-03-31   7.37
```

MATLAB:

```
    Id: 'xom'
  Date: [735234 735324 735415 735507 735599 735689]
fg_eps: [9.4700 9.7000 9.8200 7.9600 7.6600 7.3700]
```

In this example, the date argument is using relative rather than absolute dates. To specify relative dates, enter the number of periods and a period code, such as D for days, W for weeks, or Q for quarters and Y for years. When using relative dates, "0" refers to the most recent time period. Therefore, 0Q refers to the most recent quarter end, while -1Q refers to two quarters ago.

3.3.5.3 F.CorporateActionsDividends

The syntax for the F.CorporateActionsDividends function is:

```
data = F.CorporateActionsDividends(ids, start date, end date, optional arguments)
```

where,

ids	Array with a list of one or more security identifiers.
start date	Start date from which dividend data should be retrieved. Method of entering date is in MM/DD/YYYY format.
end date	End date for period during which dividend data should be retrieved. The end date field is for entering a future date for which the dividend data is accessed. It can be entered as a future date in MM/DD/YYYY format or as a number, e.g. 50, which reflects 50 days from today which is set as the end date. **Note:** When entering number of days, the maximum value that can be entered is 50.

Optional arguments,

splitadj	Allows for split adjustment to be specified. This argument must be entered as: 'splitadj','9' to retrieve unadjusted dividends.
ngflag	Specify 'ngflag ','y ' to return a flag that indicate whether the dividend rate returned is a net or gross. The output would be a G or N flag.
symbol	Argument allows for the CUSIP to be retrieved as the last column (by default SecId is the first field that is retrieved when running a CorporateActionsDividends function). This argument must be entered as 'symbol', 'y'.
curr	The optional currency argument to specify the currency in which the stock dividend data is returned.
universe	Screening expression to limit the universe
ison	Ison-codes can be used to limit the universe ISON_MSCI_WORLD(0,1) is written as ‘ison’,’msci_world’,’isonParams’,’0,1’
isonParams	The arguments within brackets in the ison-code
secId	Currently, the stat packages display the ticker by default in the first column but will now display whatever values are entered in the ids= argument. The secId=Y parameter will now be used to display whatever is

	entered in the ids= argument.
OFDB	Universe is the constituents of an OFDB file, default directory is Client, if the OFDB is stored in another location the path must be included
OFDBDate	Specific date for the constituents of the OFDB
summary	When 'summary' and 'Y' is used as an argument, it will display a more detailed view including dividend description and will group dividends paid at the same time together. This is more common for Australian securities.

Example

In this example, extract the stock dividend information for Volkswagen from 1/1/2011 up to 1 day from today.

```
data = F.CorporateActionsDividends('VOW-DE','1/1/2011','1');
```

With the output:

R:

```
  SecId      Date Currency P.DIVS   P.DIVS.TYPED P.DIVS.PAYDATE P.DIVS.RECDATE
1 VOW-DE 2011-05-04    Euro   2.2 Yearly payment    2011-05-04    2011-05-03
2 VOW-DE 2012-04-20    Euro   3.0 Yearly payment    2012-04-20    2012-04-19
3 VOW-DE 2013-04-26    Euro   3.5 Yearly payment    2013-04-26    2013-04-25
4 VOW-DE 2014-05-14    Euro   4.0 Yearly payment    2014-05-14    2014-05-13
```

MATLAB:

```
          SecId: 'VOW-DE'
           Date: [734627 734979 735350 735733]
       Currency: {'Euro'  'Euro'  'Euro'  'Euro'}
         P_DIVS: [2.2000 3 3.5000 4]
    P_DIVS_TYPED: {'Yearly payment'  'Yearly payment'  'Yearly payment'  'Yearly payment'}
  P_DIVS_PAYDATE: [734627 734979 735350 735733]
  P_DIVS_RECDATE: [734626 734978 735349 735732]
```

3.3.5.4 F.CorporateActionsSplits

The syntax for the F.CorporateActionsSplits function is:

```
data =  F.CorporateActionsSplits(ids, start date, end date, optional arguments);
```

where,

ids	Array with a list of one or more security identifiers.
start date	Start date from which split data should be retrieved. Method of entering date is in MM/DD/YYYY format.
end date	End date for period during which dividend data should be retrieved. The end date field is for entering a future date for which the split data is accessed. It can be entered as a future date in MM/DD/YYYY format or as a number, e.g. 50, which reflects 50 days from today which is set as the end date. **Note**: When entering number of days, the maximum value that can be entered is 50.

Optional arguments,

symbol	Argument allows for the CUSIP to be retrieved as the last column (by default SecId is the first field that is retrieved when running a CorporateActionsSplits function). This argument must be entered as 'symbol', 'y'.
universe	Screening expression to limit the universe
ison	Ison-codes can be used to limit the universe ISON_MSCI_WORLD(0,1) is written as 'ison','msci_world','isonParams','0,1'

isonParams	The arguments within brackets in the ison-code
secId	Currently, the stat packages display the ticker by default in the first column but will now display whatever values are entered in the ids= argument. The secId=Y parameter will now be used to display whatever is entered in the ids= argument.
OFDB	Universe is the constituents of an OFDB file, default directory is Client, if the OFDB is stored in another location the path must be included
OFDBDate	Specific date for the constituents of the OFDB

Example

In this example, extract the stock split information for Exxon Mobil from 1/1/1990 up to 1 day later from today.

```
data = F.CorporateActionsSplits('xom','1/1/1990','1');
```

With the Output:

R:

```
  SecId      Date P.SPLIT.FACTOR P.SPLIT.RATIO P.SPLIT.COMMENT
1 XOM-US 1997-04-14          0.5          2:1
2 XOM-US 2001-07-19          0.5          2:1  Split: 2 for 1
```

MATLAB:

```
          SecId: 'XOM-US'
           Date: [729494 731051]
   P_SPLIT_FACTOR: [0.5000 0.5000]
    P_SPLIT_RATIO: {'2:1'  '2:1'}
  P_SPLIT_COMMENT: {''  'Split: 2 for 1'}
```

> **Note:** The retrieved items with this function are the split factor, the split ratio and any available split comments.

3.3.5.5 F.ExtractBenchmarkDetail

The syntax for the F.ExtractBenchmarkDetail function is:

```
data = F.ExtractBenchmarkDetail(ids, items, dates, optional arguments);
```

where,

data	Variable name for the data returned
ids	Array with a list of one or more benchmark identifiers.
dates	One or more dates; Dates should be entered in start:end:freq format. (e.g. '20101215:20110115:d**')**
items	One or more items in Screening syntax, if FQL syntax is required it may be used with an underscore needs to be appended at the beginning of the code, i.e _P_PRICE

Optional arguments,

cutoff	Number of constituents to display; default displays all instances
useBTD	To control the alignment of historical stitching following a merger the useBTD parameter is used. When FactSet and a benchmark vendor make different choices in picking a surviving entity symbols can be returned as a dummy ticker to be used as a placeholder. To return the symbol as of the back test date 'useBTD','ON' should be used.

Example

In this example, the constituents for S&P 500 is being extracted, the default columns will always be available for this factlet.

```
data = F.ExtractBenchmarkDetail('SP50','','')
```

With the output:

R:

```
   BENCHMARK.ID      DATE SECURITY.ID    Weight
1          SP50 2014-07-31   03783310 3.36029915
2          SP50 2014-07-31   30231G10 2.47762116
3          SP50 2014-07-31   59491810 1.89185378
4          SP50 2014-07-31   47816010 1.65120888
5          SP50 2014-07-31   36960410 1.47063591
...
501        SP50 2014-07-31   38463710 0.01804375
```

MATLAB:

```
  BENCHMARK_ID: 'SP50'
          DATE: [1x501 double]
   SECURITY_ID: {1x501 cell}
        Weight: [1x501 double]
```

3.3.5.6 F.ExtractOFDBItem

The syntax for the F.ExtractOFDBItem function is:

```
data = F.ExtractOFDBItem(OFDB, ids, items, dates, optional arguments)
```

where,

data	Variable name for the data returned
OFDB	OFDB file from which the items should be used. The default directory is Client if other locations are used the path must be specified i.e personal:MyOFDB
ids	Array with a list of securities to extract the data for. If left blank data for all securities in the OFDB will be extracted.
dates	One or more dates; Dates should be entered in start:end:freq format. (e.g. '20101215:20110115:d')
items	One or more items from the OFDB

Optional arguments,

datesOnly	Displays only the dates that are in an OFDB with the parameter 'datesOnly','Y'
idsOnly	Displays the unique Ids that are in an OFDB with the parameter 'idsOnly','Y'
itemsOnly	Displays the items that are in an OFDB with the parameter 'itemsOnly','Y'
universe	Screening expression to limit the universe
feelback	If the feelback argument is not used, the returned data series will "feel back" over NAs to find the last actual data point and carry this data forward over the NAs. For the data not to carry forward, use 'feelback', 'N'. The data is then returned as it is in the database.
fqlflag	Optional argument that is necessary because by default, the ExtractOFDBItem factlet goes through screening, but when there are _S in the Identifier or spaces between the identifiers, it is necessary to extract the data through FQL to get the values. Need to specify 'fqlflag','y'.
cal	Calendar setting, arguments include: **FIVEDAY:** Displays Monday through Friday, regardless of whether there were trading holidays.

	FIVEDAYEOM: Displays Monday through Friday including a weekend date if it falls on the last day of the month. Where the month-end does not fall on a weekend, the calendar will act just as the standard five-day calendar. **SEVENDAY:** Displays Monday through Sunday. **AAM:** For Exchange code uses the calendar of a specific exchange, represented by the exchange code. If there is no calendar available for a specific exchange, the calendar will default to FIVEDAY.
unsplit	Displays prices with split adjustments in unsplit form.
currency	The currency in which the data is to be returned, using a string with the three character ISO code (e.g. 'USD' or 'EUR'). This will only work when "Currency Mapping" is used in the OFDB.

Example

In this example, retrieve the price and shares data uploaded into the OFDB file titled MyPortfolio for Microsoft and IBM as of 4 trading days ago, denoted with the date argument -3D.

```
data = F.ExtractOFDBItem('MyPortfolio','MSFT,IBM','PRICE,SHARES','-3D');
```

With the output:

R:

```
  Id      Date ofdb.price ofdb.shares
1 MSFT 2014-07-28       46        6000
2  IBM 2014-07-28      180        1500
```

MATLAB:

```
1x2 struct array with fields:

    Id
    Date
    ofdb_price
    ofdb_shares
```

Note: If there is an error in the name of the OFDB file or the user does not have access, the data will be returned as NAs.

3.3.5.7 F.ExtractOFDBUniverse

The syntax for the ExtractOFDBUniverse function is: (Date and fqlflag are optional)

```
data = F.ExtractOFDBUniverse('OFDB','date','DateArgument','fqlflag','Y/N');
```

where,

data	Variable name for the data returned
OFDB	OFDB file from which the items should be used. The default directory is Client if other locations are used the path must be specified i.e personal:MyOFDB
date	The date the OFDB constituents should be extracted for, only one date can be specified.
fqlflag	Optional argument that is necessary because by default, the ExtractOFDBItem factlet goes through screening, but when there are _S in the Identifier or spaces between the identifiers, it is necessary to extract the data through FQL to get the values. Need to specify 'fqlflag','y'.

Example

In this example, retrieve the securities stored as of the end of the most recent trading day in the OFDB file titled MyOFDB.

data = F.ExtractOFDBUniverse('MyOFDB');

3.3.5.8 F.ExtractScreenUniverse

The syntax for the F.ExtractScreenUniverse function is:

data = F.ExtractScreenUniverse(screen, name (optional argument));

where,

data	Variable name for the data returned
screen	Universal Screen for which the universe should be extracted. The default location is Client: for any other location the path must be specified.
name	Optional parameter to display the name of the securities extracted. Specified as **'name', 'Y'.**
All	Pulls all of the columns from a saved screen.
Backtestdate	Ability to set a backtest date dynamically within the stat packages. This requires an additional subscription to FactSet's backtesting utilities.
removeColumns	Ability to hide specific columns from being displayed in the output. Requires the use of the "All" parameter as well.
includeColumns	Ability to select specific columns to display in the output. Requires the use of the "All" parameter as well.

Example 1

In this example, retrieve the securities stored in the screen titled MyScreen. The output displays the CUSIPS for each security.

data = F.ExtractScreenUniverse('MyScreen');

Example 2

In this example, retrieve all of the securities and parameters saved in the screen. Also, set a backtest date to 6/30/2014.

Data=F.ExtractScreenUniverse('Personal:MyScreen','All','Y','Backtestdate','20140630')

Example 3

In this example, retrieve all of the securities returned by the screen, as well as only the first 3 parameters.

Data=F.ExtractScreenUniverse('Personal:MyScreen','All','Y','includeColumns','1,2,3,5,7')

3.3.5.9 F.ExtractOptionsSnapshot

The syntax for the F.ExtractOptionsSnapshot function is:

data = F.ExtractOptionsSnapshot(items, date, cond1, compval1, cond2, compval2, cond3, compval3)

where,

data	Variable name for the data returned
items	One or more items from the FactSet Option Derived database. **Note:** If the items argument is left blank the price for the underlying security will be returned for this field.
date	One or more dates; Dates should be entered in start:end:freq format. (e.g. '20101215:20110115:d'**)**
cond1/2/3	Screening condition with "=" or ">" or "<"; P_OPT_UNDERLYING_SECURITY=(default);P_OPT_ALL_VOLUME>

compval1/2/3	Value that meets cond1/2/3

Example

In this example a put or call flag, closing price, expiry date and delta is extracted for the options passing the screening conditions that FactSet (FDS) is the underlying security and the expiration date is before 20151231. Please note that the date used in the following example needs to be a future date.

```
Data=F.ExtractOptionsSnapshot('P_OPT_CALL_OR_PUT,P_OPT_CLOSE_PRICE,P_OPT_EXP_DATEN,P_OPT_DELTA','','P_OPT_UNDERLYING_SECURITY','FDS','P_OPT_EXP_DATEN<','20151231','','');
```

With the Output:

R:

```
        Id      Date P.OPT.CALL.OR.PUT P.OPT.CLOSE.PRICE P.OPT.EXP.DATEN P.OPT.DELTA
1  FDS.US#C63KF 2015-10-07              0              NaN         20151218         NaN
2  FDS.US#C8W5C 2015-10-07              0              NaN         20151218    0.950529
3  FDS.US#CB3SB 2015-10-07              0              NaN         20151218    0.949434
4  FDS.US#CPGFP 2015-10-07              0              NaN         20151218    0.931311
5  FDS.US#CBXCH 2015-10-07              0              NaN         20151218    0.931583
6  FDS.US#CM9KV 2015-10-07              0              NaN         20151218         NaN
7  FDS.US#CCH73 2015-10-07              0              NaN         20151218    0.986954
8  FDS.US#CV8X7 2015-10-07              0            27.70         20151218    0.952299
```

MATLAB:

```
1x172 struct with 6 fields:

  Id
  Date
  P_OPT_CALL_OR_PUT
  P_OPT_CLOSE_PRICE
  P_OPT_EXP_DATEN
  P_OPT_DELTA
```

3.3.5.10 F.ExtractSPARData

The syntax for the F.ExtractSPARData function is:

```
data  =  F.ExtractSPARData(ids, items, date, OFDB (optional argument));
```

where,

data	variable name for the data returned
ids	CellString array array with a list of one or more benchmarks or funds.

items	CellString array with a list of one or more FactSet data items to display for the selected benchmarks or funds
date	One or more dates; Dates should be entered in start:end:freq format. (e.g. '20101215:20110115:d**'**)
OFDB	OFDB file used to limit the universe
cal	Calendar setting, arguments include: **FIVEDAY**: Displays Monday through Friday, regardless of whether there were trading holidays. **SEVENDAY**: Displays Monday through Sunday.

Example

In this example, for the two specified Morningstar funds retrieve the fund family name and the benchmark name.

```
data = F.ExtractSPARData('MEUR:F0GBR04AWX,MEUR:F000000GJF', 'SPAR_FUND_FAMILY,SPAR_MEUR_BM_NAME1','');
```

With the Output:

R:

```
  Id      Date          spar.fund.family          spar.meur.bm.name1
1 MEUR:F0GBR04AWX 2014-08-05 Aberdeen Global Services S.A. MSCI AC Asia Pac Ex JPN NR USD
2 MEUR:F000000GJF 2014-08-05       AZ Fund Management S.A.
```

MATLAB:

```
1x2 struct array with fields:

  Id
  Date
  spar_fund_family
  spar_meur_bm_name1
```

3.3.5.11 F.ExtractVectorFormula

The syntax for the F.ExtractVectorFormula function is:

```
data  =  F.ExtractVectorFormula(ids, items, optional arguments);
```

where,

data	Variable name for the data returned
ids	CellString array with a list of one or multiple security identifiers
items	CellString array with a list of one or more FactSet data items in the FQL language

Optional arguments,

universe	Screening expression to limit the universe
ison	Ison-codes can be used to limit the universe ISON_MSCI_WORLD(0,1) is written as 'ison','msci_world','isonParams','0,1'
isonParams	The arguments within brackets in the ison-code
OFDB	Universe is the constituents of an OFDB file, default directory is Client, if the OFDB is stored in another location the path must be included
OFDBDate	Specific date for the constituents of the OFDB
combinedOutputTypes	Required argument when matrix and vector output formats are requested in the same

	call.

Example

In this example, extract the business segment sales breakdown, with labels, as of the most recent fiscal year end for IBM and GE using the FactSet Fundamentals database. The ExtractVectorFormula function is used to extract this data because the output is a row of data, and it is not indexed by Id, data item and date. Rather it is a list where IBM has 5 business segments and GE has 8 segments.

R: **data = F.ExtractVectorFormula('IBM,GE','FF_SEGMENT_RPT_DATA(ANN,0,,,,**
\'SALES\',,BUS,\'SEG\'),FF_SEGMENT_RPT_LABELS(ANN,0,,,,,BUS,\'SEG\')');

MATLAB: **data = F.ExtractVectorFormula('IBM,GE','FF_SEGMENT_RPT_DATA(ANN,0,,,,**
''SALES'',,BUS,''SEG''),FF_SEGMENT_RPT_LABELS(ANN,0,,,,,BUS,''SEG'')');

R:

	Id	ff.segment.rpt.data	ff.segment.rpt.labels
1	IBM	20131231	
2	IBM	38551	Global Technology Services
3	IBM	25932	Software
4	IBM	18396	Global Business Services
5	IBM	14371	Systems & Technology
6	IBM	2022	Global Financing
7	GE	20131231	
8	GE	42917	General Electric Capital
9	GE	23777	Power & Water
10	GE	21411	Aviation
11	GE	18186	Healthcare
12	GE	16615	Oil & Gas
13	GE	8313	Appliances & Lighting
14	GE	6721	Energy Management
15	GE	5873	Transportation

MATLAB:

```
1x2 struct array with fields:

    Id
    ff_segment_rpt_data
    ff_segment_rpt_labels
```

> **Note:** Single quotes in an FQL formula needs to be escaped by a backslash (R) or single quote (MATLAB).

3.3.5.12 F.ExtractEconData

The syntax for the F.ExtractEconData function is:

data = F.ExtractEconData(ids, items, optional arguments**);**

where,

data	Variable name for the data returned

ids	CellString array with a list of the country identifiers when used for the standardized economic database only, if other databases the ids argument should be left blank.
items	CellString array with a list of one or more FactSet data items from the Economic database

Optional arguments,

date	One or more dates; Dates should be entered in start:end:freq format. (e.g. '20101215:20110115:d')
NFB	NFB is the optional "no feel back" argument in FQL codes. If you do not use the NFB argument, the returned data series will contain NAs where the data is not available (default is NFB=1). If you want the data to "feel back" over NAs to find the last actual data point and carry this data forward, set the NFB argument to either 0 or 2.
TSName	Used to display the time series value of the item in the label of the column where the data is being displayed. Ie SPEC_ID_DATA('WTI-FDS:FG_PRICE',-121,-1,M) is displayed in the column label as WTI-FDS. Specified as 'TSName','Y'
decimals	Positionally set according to the items in the selection, ie 'decimals','',,3,4,3'

Example

This example retrieves industrial production data for the United States using the FactSet Economics database, starting 122 months ago (denoted with -121) until two months ago (denoted with -1).

R: **data = F.ExtractEconData('','FDS_ECON_DATA(\'FRBIPSB50001\',-121,-1,M,STEP,AVERAGE,1)');**

MATLAB: **data = F.ExtractEconData('','FDS_ECON_DATA(''FRBIPSB50001'',-121,-1,M,STEP,AVERAGE,1)');**
With the output:

R:

```
          Id      Date F.econ.data
1  fds_econ_data 2004-06-30     91.7493
2  fds_econ_data 2004-07-30     92.4549
3  fds_econ_data 2004-08-31     92.4832
4  fds_econ_data 2004-09-30     92.5476
5  fds_econ_data 2004-10-29     93.4371
...
121 fds_econ_data 2014-06-30     103.9152
```

MATLAB:

```
        Id: 'fds_econ_data'
        Date: [1x121 double]
  fds_econ_data: [1x121 double]
```

3.3.5.13 F.ExtractAlphaTestingSnapshot

The syntax for the F.ExtractAlphaTestingSnapshot function is:

```
data = F.ExtractAlphaTestingSnapshot(useStat,headers,model,report,
items,security,date,resultType,sortOrder,sortCol,reportSettingName);
```

> **Note:** The ExtractAlphaTestingSnapshot function is used for extracting model results that use the Alpha Testing codes AT3_RESULT_DATA or AT3_RESULT_STAT.

where,

data	Variable name for the data returned
useStat	Blank or N. Leave a blank in quotes ('') to extract the main report data and extracted with the

	code AT3_RESULT_STAT. Specify (N) to extract the constituent data for the report, with each security/period in each row and each data item result in each column. This includes the raw, universe return and fractile data to display the raw data available for the companies in the specified universe, compared to the data available if outlier limitations are set within the model, along with the fractile values. This company level data is extracted with the code AT3_RESULT_DATA and goes into the aggregate calculation extracted with AT3_RESULT_STAT.
headers	Y or N. Specify if headers are required.
model	String specifying a AT3 model. Format as client:model name.
report	Name of the report to be extracted, i.e. CONSITUENTS, FRACTILES or PERIODS etc.
items	CellString specifying items (headers) or column numbers, ALL will return all items in report.
security	A single security can be specified.
date	A single date can be specified.
resultType	M or S. Main or Summary data respectively. Defaults to M.
sortOrder	A or D. Displays data in either Ascending or descending order.
sortCol	Column from which to sort the data
reportSettingName	String specifying name of the report setting or template.

Example

In this example, extract column 6 and 7 from the Constituents report sorted by column 6 of the Alpha Testing model titled Calculation Example Model.

```
data = F.ExtractAlphaTestingSnapshot('N','N','Factset:Calculation Example Model','CONSTITUENTS','6,7','','','','D','6','');
```

With the output:

R:

```
Identifier  Company.Name    Periods   Weight Market.Capitalization
1    G8865370     Wellco Enterprises Inc.  2004-05-28   0.792022     28015056.000
2    G8865312     Wacoal Holdings Corp.    2004-05-28   0.729555     13745582.000
3    94947610     WACHOVIA [...]S&P500 09    2004-05-28   0.714043     13558773.000
4    93000420     VILLAGE [...]/2007        2004-05-28   0.510412     13257467.000
5    92990360     VI GROUP SPONSORED ADR   2004-05-28   0.384092     5713187.000
...
```

MATLAB:

```
1x60 struct array with fields:

    Identifier
    Company_Name
    Periods
    Weight
    Market_Capitalization
```

3.3.5.14 F.LSD_Ownership

The syntax for the F.LSD_Ownership function is:

```
data = F.LSD_Ownership(ids, items, optional arguments);
```

where,

data	Variable name for the data returned
ids	CellString array with one or more identifiers for securities or holders.
items	CellString array with a list of one or more FactSet data items from the Lionshares Database

Optional arguments,

combinedOutputTypes	Required argument when matrix and vector output formats are requested in the same call.

Example

In this example, extract the names in English of the top 3 institutional (signified by the F in the request code) holders (signified by the H in the request code) for Apple using the code LSD_NAME_TOP_HLDR.

R: **data = F.LSD_Ownership('AAPL-US', 'OS_TOP_HLDR_NAME(3,0D,,MTD,,F,SEC,\\"EN\\")')**

MATLAB: **data = F.LSD_Ownership('AAPL-US', 'OS_TOP_HLDR_NAME(3,0D,,MTD,,F,SEC,"EN")')**

With the output:

R:

```
  Id         lsd.name.top.hldr
1 aapl    The Vanguard Group, Inc.
2 aapl SSgA Funds Management, Inc.
3 aapl     BlackRock Fund Advisors
```

MATLAB:

```
            Id: 'aapl'
  lsd_name_top_hldr: {'The Vanguard Group, Inc.'  'SSgA Funds Management, Inc.'  'BlackRock Fund Advisors'}
```

3.3.5.15 F.EstimatesOnDemand

The syntax for the F.EstimatesOnDemand function is:

data = F.EstimatesOnDemand(ids, items, report, startDate, optional arguments**)**

where,

data	variable name for the data returned
ids	CellString array with a list of one or multiple security identifiers
items	CellString array with a list of one or more FactSet data items from the FactSet Estimates database (e.g., EPS, Sales, Net Debt). **Note:** Table 1 in Appendix has a comprehensive list of items for which estimates are available using this function.
report	Allows specification of the types of estimates report through which the data is retrieved. The available reports are: Actuals, BrokerDetail, BrokerSnapshot, Consensus, Guidance, Surprise, ConsensusReco, DetailedReco, and Coverage.
startDate	The start date as of which the estimate data is retrieved.

Optional arguments,

end	The end date as of which the estimate data is retrieved
freq	The frequency of which the estimate data is retrieved

fiscalPeriod	The fiscal period for the estimate item. The option is available of looking at historical, current, or future fiscal periods. The fiscal period can be specified using relative dates. The arguments entered as relative dates represent a date relative to the most recently updated period. For example, 0 (zero) represents the most recently reported period; -1 represents the time period prior to the most recently reported period. Arguments entered can be -1, 0, 1, 2, etc.
periodType	The argument can be entered as “annual”, “quarterly”, or “semi”, depending on the type of estimates data request. Not all equities have estimates for all period types.
fields	Allows for specification of a select number of fields to extract. **Note:** Each section provides a detailed list of the output fields associated with each FactSet Estimates report.
timeStamp	Display the publication time associated with the publication date. The argument would be set up as: 'timestamp', 'y' and it can be used with an actuals report.
reportDate	Display report date. The argument would be set up as: 'reportDate', 'y' and can be used with the Broker Detail report.
previousDates	Used with the Consensus report and refers to previous date as of which estimates can be retrieved and compared to the estimates retrieved as of the date argument. For example, if EPS estimates are displayed as of now, allows clients to compare the EPS estimates as of i.e. 30 days ago.
prev	If the previousDates argument is used the ‘fields’ and ‘prev’ should be appended.
display	Used with the Broker Detail report. If utilizing HISTO for the historical look an ‘end date’ argument must be entered. If utilizing the SNAP mode, an ‘end date’ parameter is not needed unless looking for the current consensus less than 100 days old. Otherwise SNAP will bring back the current consensus as of the last 100 days.
statistic	Used with the Surprise report. There are a number of different statistics that the client can bring back using the Surprise Report. They have the ability to specify which one they prefer. The list includes: Mean, Median (MED), High Estimate (HIGH), Low Estimate (LOW), Sigma and Standard Deviation (STDDEV).
Offset1/offset2	Used with the Surprise report. This parameter is to change the number of days used before and after the report date to calculate price impact. The argument would be set up as: 'offset1', 'offset2'.
Currency	Allows all values to be changed to the specified currency. By default, the currency is the value of the security.
meanText	Displays the Rating Name. The argument would be set up as: 'meanText', 'y' and can be used with the Consensus Recommendation report.
estCurrency	In cases where the security’s local currency does not match the Currency of the estimates the argument 'estCurrency', 'Y' can be used. This changes the currency field to display the Estimate Currency of the first security returned. Therefore, if the first security in the list as an Estimate Currency of EUR, all of the results will return in EUR. Also, the field heading changes to EST_CURRENCY.
showExcluded	Available for BrokerDetail and BrokerSnapshot, specifying this to N will only display the broker estimates that are included in the consensus; default is to show all values.
universe	Screening expression to limit the universe
ison	Ison-codes can be used to limit the universe ISON_MSCI_WORLD(0,1) is written as 'ison', 'msci_world', 'isonParams', '0,1'
isonParams	The arguments within brackets in the ison-code
OFDB	Universe is the constituents of an OFDB file, default directory is Client, if the OFDB is stored in another location the path must be included
OFDBDate	Specific date for the constituents of the OFDB
cal	Calendar setting, arguments include: **FIVEDAY:** Displays Monday through Friday, regardless of whether there were trading holidays. **SEVENDAY:** Displays Monday through Sunday.
entry_datetime	Allows a user to screen out the results based on the entry_datetime field in broker detail. Ex, entry_datetime=2/5/2015.
act	Can be set to “HIDE” to add the ability to hide broker actuals in a consensus report. Ex, act=”hide”
estdate	Can set the estdate to Input
pev	Can set to “Y” to use Post Event Consensus
RT	Can set to “Y” to disable realtime
FixedRate	Can set to “Y” to set a fixed exchange rate.

Example

In this example, retrieve for Microsoft the EPS and sales actuals as of now for the current reported fiscal year that the company is in (denoted with the fiscal period 0 argument in the request syntax below) and the values for the fiscal year before that (denoted with the -1 fiscal period argument).

```
data = F.EstimatesOnDemand('msft','eps,sales','actuals','NOW', 'fiscalperiod','-1,0','periodtype','annual');
```

With the output:

R:

```
 SecId CURRENCY FE.ITEM FE.PER.REL FE.REPORT.FY  PUBDATE    DATE   FE.ACTUAL  FE.ACTUAL.FLAG
1  MSFT     USD    EPS       -1  2013-07-19 2013-07-18 2013-06-30     2.58       1
2  MSFT     USD    EPS        0  2014-07-23 2014-07-22 2014-06-30     2.63       1
3  MSFT     USD  SALES       -1  2013-07-19 2013-07-18 2013-06-30 77849.00       1
4  MSFT     USD  SALES        0  2014-07-23 2014-07-22 2014-06-30 86833.00       1
```

MATLAB:

```
          SecId: 'MSFT'
       CURRENCY: {'USD'  'USD'  'USD'  'USD'}
        FE_ITEM: {'EPS'  'EPS'  'SALES'  'SALES'}
     FE_PER_REL: [-1 0 -1 0]
   FE_REPORT_FY: [735434 735803 735434 735803]
      PUBDATE: [735433 735802 735433 735802]
         DATE: [735415 735780 735415 735780]
    FE_ACTUAL: [2.5800 2.6300 77849 86833]
  FE_ACTUAL_FLAG: [1 1 1 1]
```

3.3.5.16 F.TickHistory

Note: *Requires FactSet plugin version 3.1+*

The syntax for the F.TickHistory function is:

```
Data=F.TickHistory('IDs','StartDate','StartTime','EndDate','EndTime','Interval','fields','OptionalFields')
```

Where,

Data	Variable name for the data returned
IDs	Identifier for which data will be returned. Maximum of one identifier.
StartDate	Start Date of request in format YYYYMMDD
StartTime	Start Time of request in format HHMMSS
EndDate	End Date of request in format YYYYMMDD
EndTime	End Time of request in format HHMMSS
Interval	Frequency at which data will be returned
Fields	Hardcoded argument to allow for specific field requests

Optional Fields

BID_1	The last bid price or last bid price in an interval
BID_VOL_1	The volume of the last bid or last bid in an interval
BID_EXCH_1	The exchange of the last bid or last bid in an interval
ASK_1	The last ask price or last ask price in an interval
ASK_VOL_1	The volume of the last ask or last ask in an interval

ASK_EXCH_1	The exchange of the last ask or last ask in an interval
LAST_1	The last trade price or last trade price in an interval
LAST_DATE_1	The date of the last trade or last trade in an interval
LAST_TIME_1	The time of the last trade or last trade in an interval
LAST_VOL_1	The volume of the last trade or last trade in an interval
LAST_EXCH_1	The exchange of the last trade or last trade in an interval
CUM_VOL	The daily cumulative volume or the cumulative volume for trades inside an interval
VWAP	The volume weighted average price for the trades inside an interval
OPEN_1	The first trade of an interval
HIGH_1	The highest trade price in an interval
LOW_1	The lowest trade price in an interval
TRADE_CONDITION	Trade condition of the last trade or last trade price in an interval
GMT_OFFSET	GMT Offset in Minutes
PRICE_CURRENCY	Currency of Quotes and Prices

Intervals

Interval	Description	Max Days Requested
0	Every Tick	1
1S	1 Second	1
5S	5 Seconds	1
10S	10 Seconds	1
15S	15 Seconds	1
30S	30 Seconds	15
1M	1 Minute	30
2M	2 Minutes	60
5M	5 Minutes	60
10M	10 Minutes	60
15M	15 Minutes	60
30M	30 Minutes	60
1H	1 Hour	60

Limitations and Notes

- If the number of days requested exceeds the “Max Days Requested” listed above, MATLAB and R will only return data for 1 day.
- A maximum of 10 TickHistory requests can be made in any one-minute period.
- If more than 100,000 rows of data are requested in any one-minute period, the next request will be delayed until the next minute.
- Only one ticker may be specified in each request

Example

In this example, retrieve the Last Price, Last Time and Last Date for IBM every minute for the day of 9/25/2014.

```
Data=F.TickHistory('IBM','20140925','093000','20140925','160000','1M','fields','LAST_1,LAST_TIME_1,LAST_DATE_1')
```

With the output:

R:

```
   LAST_1 LAST_TIME_1 LAST_DATE_1
1  192.270   93055968   20140925
2  191.930   93159708   20140925
3  191.900   93257886   20140925
4  191.910   93359856   20140925
5  191.970   93446543   20140925
6  192.120   93548632   20140925
...
```

MATLAB:

```
      LAST_1: [1x390 double]
  LAST_TIME_1: [1x390 double]
  LAST_DATE_1: [1x390 double]
```

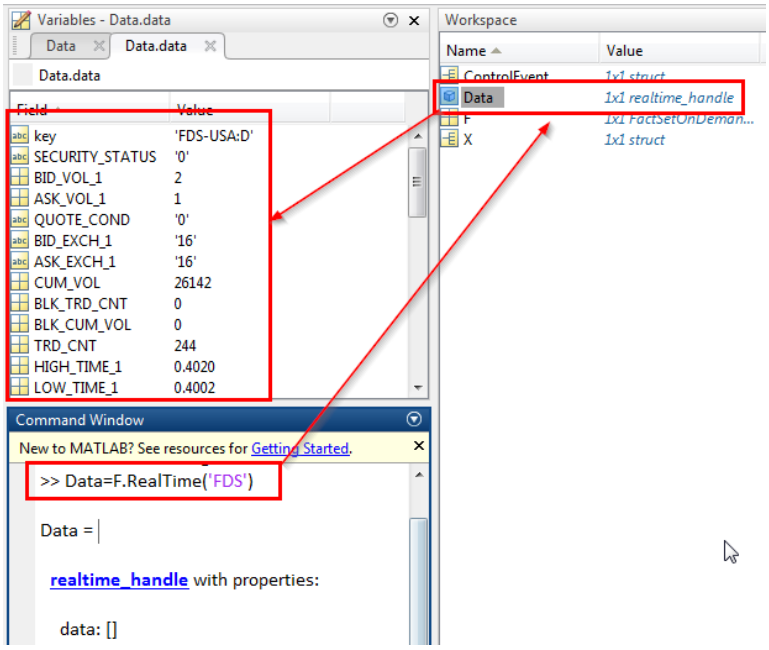
3.3.5.17 F.RealTime

The realtime function allows users to stream realtime exchange data. Multiple securities can be streamed at once and the streaming data can be referenced through the variables that are created. The FactSet Workstation must be running for this function to work and the FactSet Datafeed Toolkit must be installed when using MATLAB.

Note: *Requires an additional subscription and FactSet plugin version 3.0+ for MATLAB and 3.1+ for R/Developer's Toolkit*

http://www.factset.com/download/statlink/Docs%203.0/Streaming

Sample MATLAB Results



Sample R Syntax and Results

Data=F.RealTime('FDS') , where "FDS" is the ticker.

R Results:

```
> Data=F.RealTime('FDS')
Pulled realtime data for: FDS
> Data
        Key SECURITY_TYPE PRODUCT PRICE_CURRENCY SECURITY_STATUS HALT_INFO    CUSIP
1 FDS-USA:D             1    9001            USD               0         0 30307510 FactSet
  BLK_TRD_CNT BLK_CUM_VOL TRD_CNT EXCHANGE  HIGH_TIME_1   LOW_TIME_1  LAST_TIME_1 LAST_POST
1           0           0     252    11110 09:38:53.000 09:36:16.000 10:37:51.711
    BID_TIME_1   ASK_TIME_1 FINANCIAL_STATUS LAST_VOL_1 LAST_TICK_1 LAST_EXCH_1 GMT_OFFSET
1 10:38:00.052 10:38:00.052                4        100           4           3       -300
>
```

Note: There is no streaming interface in R. Results must be pulled by re-running the F.RealTime command.

3.3.5.17 F.Snapshot

Note: *Requires an additional subscription and FactSet plugin version 3.1+*

The Snapshot service provides access to streaming exchange data and allows the "snap" of real-time prices at the user's request. When the command is run, a list of all available exchange data items will be returned by default.

Syntax:

Data=F.Snapshot('Ticker')

Requests can be made for multiple tickers as well in a comma delimited format.

Data=F.Snapshot('Ticker1,Ticker2,Ticker3')

Fields can be selectively returned at a user's request to limit the results.

Data=F.Snapshot('Ticker','Fields','Fields Requested')

Sample Request with Result (Single Ticker)

Data=F.Snapshot('FDS')

Data =

REQ_SYM: {'FDS'}
KEY: {'FDS-USA:D'}
EXCHANGE: 11099
ORIG_SEQUENCE: 1525089
BID_1: 155.1800
BID_DATE_1: 20150218
BID_TIME_1: 135757694
BID_VOL_1: 1
BID_TICK_1: 1
BID_EXCH_1: 1
BID_EXCH: 11100
ASK_1: 155.3300
ASK_DATE_1: 20150218
LAST_EXCH_1: 3
CLOSE_1: 154.0400
CLOSE_TIME_1: 161507000
LAST_EXCH: 10050
LAST_UNOFF_1: 155.2800
LAST_UNOFF_DATE_1: 20150218
LAST_UNOFF_TIME_1: 135711897
LAST_UNOFF_VOL_1: 30
LAST_UNOFF_COND_1: {'01 2 94 24'}
LAST_UNOFF_EXCH_1: 14
LAST_PREMKT_1: 0
LAST_PREMKT_TIME_1: 0
LAST_PREMKT_VOL_1: 0
LAST_UNOFF_EXCH: 10075
LAST_PREMKT_CUM_VOL: 0
UPPER_TRADING_BAND: 163.0600
NYSE_BUY_IMBALANCE: 0
NYSE_SELL_IMBALANCE: 0
NAS_BUY_IMBALANCE: 0
NAS_SELL_IMBALANCE: 0
OPEN_1: 153.7300
HIGH_1: 155.7000
HIGH_TIME_1: 120617000
LOW_1: 153.0102
LOW_TIME_1: 93217000
VENUE: {'FINN'}
CORR_LAST: NaN
CORR_LAST_VOL: NaN
SECURITY_TYPE: 1
CUSIP: 30307510

ASK_TIME_1: 135757694
ASK_VOL_1: 1
ASK_EXCH_1: 9
ASK_EXCH: 11103
SHORT_SALE_INDICATOR: 0
QUOTE_COND: 0
LAST_1: 155.3700
LAST_DATE_1: 20150218
LAST_TIME_1: 135344317
LAST_VOL_1: 30
LAST_COND_1: 6
LAST_TICK_1: 3

LAST_POSTMKT_1: 0
LAST_POSTMKT_TIME_1: 0
LAST_POSTMKT_VOL_1: 0
LAST_POSTMKT_CUM_VOL: 0
OFFBOOK_CUM_VOL: 0
CUM_VOL: 84347
TURNOVER: 1.3060e+04
VWAP: 154.8462
TRD_CNT: 1284
BLK_TRD_CNT: 0
BLK_CUM_VOL: 0
PREV_CLOSE: 154.0400
PREV_CLOSE_DATE: 20150217
LOWER_TRADING_BAND: 147.5300

DESCRIPTION: {'FactSet Research Systems Inc.'}
SHARES_OUTSTANDING: 41.7070
PRICE_CURRENCY: {'USD'}
SECURITY_STATUS: 0
GMT_OFFSET: -300
FINANCIAL_STATUS: 4
HALT_INFO: 0
AVG_30DAY_VOL: 0.2473
AVG_5DAY_VOL: 0.2121
TRADE_CONDITION: {'01 2 94 24'}

Sample Request with Result (Selected Fields)

```
Data=F.Snapshot('FDS','Fields','PREV_CLOSE,VWAP,HIGH_1,LOW_1')
```

Data =

REQ_SYM: {'FDS'}
KEY: {'FDS-USA:D'}
PREV_CLOSE: 154.0400
VWAP: 154.8709
HIGH_1: 155.7000
LOW_1: 153.0102

3.3.5.17 F.Documents

Note: *Requires an additional subscription and FactSet plugin version 3.1+*

The Documents service provides access for the retrieval of news stories, investment research reports, filings, and transcripts. When requested, summary information of the documents will be returned, including an http URL to access the resulting documents.

Note: This service requires an additional subscription to access this data.

There are 5 Documents functions:

Function	Description
F.DocumentsSearch()	This function returns headlines and http URL's, linking to the documents returned by the search.
F.DocumentsCount()	This function returns the count of headlines returned by the search, grouped by source.
F.DocumentsSources()	This request returns a list of possible sources that can be used as arguments in the "Search" function.
F.DocumentsCategories()	This request returns a list of possible categories that can be used as arguments in the "Search" function.
F.DocumentsTimezones()	This request returns a list of possible time zones that can be used as arguments in the "Search" function.

Search Arguments

Note: The only functions above that require arguments are the "Search" and "Count" reports. They both accept the same arguments and the syntax can be seen below:

Search and Count Arguments	
Argument	**Description**
Ids	Requested symbols or securities. This is a comma-separated list with a maximum of 1000. Each symbol can be a FactSet exchange symbol, CUSIP, or SEDOL. e.g. Ids=IBM,GOOG-USA,GSK-GB
sd	Start Date – format is YYYYMMDD or relative +/- days (0,-1,etc) e.g. SD=20121221 or SD=0 **Default = today (0)** **Note – this parameter is ignored if the relative time (rt) parameter is used.**
ed	End Date – format is YYYYMMDD or relative +/- days (0,-1,etc) e.g. ED=20121221 or ED=-10 **Default = today (0)** **Note – this parameter is ignored if the relative time (rt) parameter is used.**
st	Start Time – format is HHMMSS e.g. ST=09000 **Default = 000000** **Note – this parameter is ignored if the relative time (rt) parameter is used.**
et	End Time – Format is HHMMSS e.g. ET=230000 **Default = 235959** **Note – this parameter is ignored if the relative time (rt) parameter is used.**
rt	Relative Tme – format is –HHMMSS RT is always between the time specified and "now". e.g. rt=-120000 (fetch headlines from between 12 hours ago and now) When using RT, all of the other date and time parameters (ST, ET) will be ignored 24 hours is the maximum historical time to specify (-235959).
N	Number of Results to return (SEARCH only) Maximum Limit = 1000 (for more results, combine with the PAGE parameter) e.g N=40 **Default = 25**
page	Page Number of Results to return (SEARCH only) Limit = 50 e.g. Page=2 **Default = 1**
timezone	Time zone to return story dates and times. e.g. Timezone=<field> Use timezones from F.DocumentsTimezones() Time zones are automatically adjusted for daylight savings. No toggling option is available. **Default = "America/New_York"**
sources	Code for document source to include. This is a comma-separated list. Use sources from F.DocumentsSources() e.g. Sources=SA,EDG **Default = ALL** **Note: This will be all document sources the client is permissioned to receive, not all available document sources thru FactSet.**
categories	Code for categories to include. This is a comma-separated list. Use categories from F.DocumentsCategories() e.g. categories=SB:ANLC,IN:OIL **Default = ALL**
primary_id	Type of Identifier Search to do *Y – will only return headlines of stories that have the search ID(s) as the Primary Id* *N – will return headlines of stories that mention/refer to the ID(s)* **Default = N**
search_text	Restricts the search to include any document stories which include the text searched.

	e.g. search_text=GREECE For a full listing of Boolen Text Search Operators, see Appendix C.
sym_type	Specifies the format of the symbol(s) returned in the ALL_IDS field. e.g. sym_type=sedol Values = cusip, sedol, isin, entity_id, tick, tick_region, If this parameter is not used, default symbols will result in the ALL_IDS field. Note – agreements with CUSIP and SEDOL are required to receive these symbol types.

Specialized Arguments		
Argument		**Description**
StreetAccount Specific	**sa_categories**	Code for StreetAccount specific categories to include. This is a comma-separated list. e.g. sa_categories=98,106 **Default = ALL**
Investment Research Specific	**industries**	One or more FactSet industry codes separated by commas. For company reports, the industry code represents the industry of the company that the report is about. For industry reports, the industry code represents the industry that the report is about. Industry codes are not particularly relevant for other types of documents (e.g., economic reports).
Investment Research Specific	**contributors**	One or more contributor codes. A list of contributors can be obtained via downloading a file from an FTP site at FactSet. This process will be outlined at the time of account setup. See section 2.3 for more information.
Investment Research Specific	**authors**	One or more analyst codes of authors who were the primary or secondary authors of the documents.

Sample Requests

Requests with No Arguments

Data=F.DocumentsSources()

```
> F.DocumentsSources()
Downloading Data.......
        Name                                  Value
1        EDG                                  EDGAR
2       FBLK                 FactSet Blackline Report
3       FCST           FactSet CallStreet Transcripts
4     FCSTEV FactSet CallStreet - Events Subscription
```

Data=F.DocumentsCategories()

```
> F.DocumentsCategories()
Downloading Data.
         Category           Name                              Value
1        Subjects       SB:ANLCH                 Analysts Revisions
2        Subjects      SB:BNKRPT                         Bankruptcy
3        Subjects        SB:COMM                        Commodities
4        Subjects      SB:COARLS Company Announcements and Releases
5        Subjects       SB:CORPS                  Corporate Actions
```

Data=F.DocumentsTimezones()

```
> > F.DocumentsTimezones()
Downloading Data.
          Key                          Value
1    Timezones                            ACT
2    Timezones                Africa/Abidjan
3    Timezones                  Africa/Accra
4    Timezones             Africa/Addis_Ababa
5    Timezones                Africa/Algiers
6    Timezones                 Africa/Asmara
```

Search and Count Functions

Data=F.DocumentsSearch('SD','0','ED','-2','IDS','FDS','N','5','sources','EDG')

```
Data =

     Symbol: 'FDS'
   HEADLINE: {1x2 cell}
     SOURCE: {'EDG'  'EDG'}
    ALL_IDS: {'FDS,FDS-US,'  'FDS,FDS-US,'}
  STORY_DATE: {'20150217'  '20150217'}
  STORY_TIME: {'165339'  '114746'}
  CATEGORIES: {1x2 cell}
  SEARCH_IDS: {'FDS'  'FDS'}
      LINK1: {1x2 cell}
```

Data=F.DocumentsSearch('SD','0','ED','-2','IDS','FDS','N','5','sources','EDG','categories','CN:US')

```
>> Data=F.DocumentsSearch('SD','0','ED','-2','IDS','FDS','N','5','sources','EDG','categories','CN:US')
Downloading Data...

Data =

     Symbol: 'FDS'
   HEADLINE: {1x2 cell}
     SOURCE: {'EDG'  'EDG'}
    ALL_IDS: {'FDS,FDS-US,'  'FDS,FDS-US,'}
  STORY_DATE: {'20150217'  '20150217'}
  STORY_TIME: {'165339'  '114746'}
  CATEGORIES: {1x2 cell}
  SEARCH_IDS: {'FDS'  'FDS'}
      LINK1: {1x2 cell}
```

Data=F.DocumentsCount('SD','0','ED','-2','IDS','FDS','N','5','categories','CN:US')

```
>> Data=F.DocumentsCount('SD','0','ED','-2','IDS','FDS','N','5','categories','CN:US')
Downloading Data..

Data =

   Symbol: 'FDS'
   SOURCE: {1x15 cell}
    COUNT: {1x15 cell}

>> [Data.SOURCE,Data.COUNT]

ans =

  Columns 1 through 8

   'FBLK'   'FSWA'   'NEWS_ASXD'   'HKEX'   'UKW'   'SDR'   'FCSTEV'   'FFW'

  Columns 9 through 17

   'EDG'   'FFR'   'FCST'   'FCSTTS'   'FIO'   'NEWS_FCIS'   'SA'   '0'   '0'

  Columns 18 through 29

   '0'   '0'   '0'   '0'   '0'   '0'   '2'   '0'   '0'   '0'   '0'   '0'

  Column 30

   '0'
```

Special Use Case Scenarios

Upload Data into an OFDB

The UploadToOFDB functionality allows clients to upload data into an OFDB file stored in Data Central in the FactSet workstation.

OFDB, which stands for Open FactSet Database, is a high-performance multi-dimensional database system used to securely store proprietary numeric and textual data on FactSet. OFDB is ideal for users who manage large portfolios or maintain extensive historical proprietary databases. OFDB optimizes large, multi-dimensional databases, giving FactSet users highly flexible, fast access to large volumes of complex data that can be used in many different applications.

> **Note:** The optimal use of the UploadToOFDB functionality from MATLAB or R is for ad-hoc and smaller scale data uploads and would not replace a client's needs for FTP processes or production services, for larger scale or holdings uploads into FactSet.

Requirements for UploadToOFDB

The following are the necessary requirements to upload data into an OFDB:

- Data set must have at least ID, Date and Items field;
- Fields uploaded can be iterated of any frequency or non-iterated;
- Date types can be High Precision, Integer, or Text;
- MATLAB - Dates need to be uploaded as integers in YYYYMMDD format for MATLAB or a MATLAB native date format
- R – Dates can be uploaded as yyyymmdd and mm/dd/yyyy formats.
- Data Central subscription in the FactSet workstation is necessary;

- Basic data storage access available to all clients with a premium FactSet workstation in Data Central is 1GB of storage space. Additional data storage is available and should be discussed with a FactSet sales representative.
- If attempting to upload data to a full OFDB file and thus exceeding data storage space access, there will be an error message, "Client Data Space is Full", after running an upload from MATLAB/R.

+ FactSet does not need to be launched when uploading data into an OFDB.

Creating a New OFDB

The following details are regarding the behavior of an OFDB file that is created through UploadToOFDB:

+ If the OFDB does not already exist, it will be created.
+ OFDBs created by UploadToOFDB have all fields iterated with Daily Frequency and data type High Precision for numbers and Text(32) for strings.
+ OFDB schemas define the database. If another schema is required for the OFDB file, it should first be created in Data Central. For more details regarding creating or editing schemas refer to Online Assistant page 11502.
+ Once an OFDB is created, no changes to the schema can be made through the UploadToOFDB functionality. A new OFDB would need to be created to make the necessary changes.

Modifying an Existing OFDB

The following details are regarding modifying an OFDB file through UploadToOFDB, when adding additional dates or values to that file:

+ Data for an additional date can be appended for existing securities in an OFDB file.
+ Data for additional securities can be appended to an OFDB file for the existing dates in the file or for a new date range.
+ The headers of the data uploaded must match the existing column names.
+ New Data items cannot be appended to an existing OFDB through the UploadtoOFDB functionality.

F.UploadToOFDB Syntax

Before uploading data into an OFDB file, it is necessary to first create a structure, similar to the structure of the results returned by a factlet request.

The syntax for the F.UploadToOFDB functionality is:

```
F.UploadToOFDB(OFDB, data)
```

where,

OFDB	The name of the OFDB file to which the data is getting uploaded (default directory is Personal, for other locations the path must be specified).
data	The Data structure that is uploaded to the OFDB.

MATLAB Example:

In this example a MATLAB data structure is created, using price and volume projections for the securities FDS and XOM, for the dates 4/10/2014-4/12/2014.

```
data_up(1).Id = 'FDS';
data_up(1).Date = [datenum('4/10/13') datenum('4/11/13') datenum('4/12/13')];
data_up(1).my_price = [93.23 nan 92.65];
```

```
data_up(1).my_volume = [388.205 0 359.396];
data_up(2).Id = 'XOM';
data_up(2).Date = [datenum('4/10/13') datenum('4/11/13') datenum('4/12/13')];
data_up(2).my_price = [88.68 nan 89.22];
data_up(2).my_volume = [14826.0 0 14980.0];
```

Once the MATLAB structure is created, using the UploadToOFDB syntax, the data structure titled data_up is uploaded to an OFDB file titled Analyst Forecast, saved in the Personal directory.

```
F.UploadToOFDB('Analyst_Forecast',data_up);
```

R Example:

In this example an R data frame is created, using price and volume projections for the securities FDS and XOM, for the dates 4/10/2014-4/12/2014.

```
dates = c('4/10/14','4/11/14','4/12/14')
my_price = c(93.23,NA,92.65)
my_volume = c(388.205,0,359.396)
fds_data  = data.frame(Id='FDS',Date=dates,my_price=my_price,my_volume=my_volume)
my_price = c(88.68,NA,89.22)
my_volume = c(14826.0,0,14989.0)
xom_data = data.frame(Id='XOM',Date=dates,my_price=my_price,my_volume=my_volume)
data_up  = rbind(fds_data,xom_data)
```

The R data frame is uploaded into an OFDB file titled Analyst_Forecast.

```
F.UploadToOFDB('Analyst_Forecast',data_up)
```

FactSet's Supply Chain Data

FactSet's supply chain data is used to view the complex networks of companies' key customers, suppliers, competitors, and strategic partners, including both direct (named by company) and reverse (named by other companies) relationships in your custom reports and templates with select formulas.

A subscription to FactSet Supply Chain data is required to retrieve company relationship data.

To access this data through ExtractVectorFormula, the FQL code FF_COMPANY_RELATIONSHIP is used. This returns relationship companies and related information for a given company based on your selections.

Note: FactSet's Supply Chain data requires an additional subscription. Please contact your local support team to gain access.

Syntax:

FF_COMPANY_RELATIONSHIP(relationship_type,company_type,identifier_output,data_output,relationship_direction)

Where:

Relationship_Type can be specified as:

- Competitors (COMP)
- Suppliers (SUPL)
- Clients/Customers (CUST)
- Partners (PRTR)

Company_Type can be specified as:

- Public Companies Only (PUB)
- Private Companies Only (PVT)
- Public and Private Companies (ALL)

Identifier_Type can be specified as:

- FactSet Entity Identifier (Public or Private Companies) (FEID)
- Regional Ticker (Public Companies Only) (TICKER)
- CUSIP/SEDOL (Public Companies Only) (CUSIP)
- CUSIP/SEDOL/FactSet Entity Identifier (Public and Private Companies Only) (CUSIP_FEID)
- Regional Ticker/FactSet Entity Identifier (Public and Private Companies Only) (TICKER_FEID)

Data_Output can be specified as:

- Identifier Only (IDONLY)
- Identifier and Name (IDNAME)
- All Data (ALL)

Relationship_Direction can be specified as:

- All (ALL)
- Direct (CDEF)
- Reverse (ODEF)

Note: Competitor and Partner relationships are the same type of stated relationship for any direction. “Other” Customer relationships are when the other company mentions the source company as a distinct supplier. "Other" Supplier relationships are when the other company mentions the source company as a distinct customer

More detailed information for FF_COMPANY_RELATIONSHIP is available on OA 17503

Example 1:

In this example the below earnings statement from Mattel is investigated through the FF_COMPANY_RELATIONSHIP code.

“On Thursday July 17th, Mattel is scheduled to report its earnings. Mattel's relationship with Walmart accounts for 18% of its total Barbie doll and Hot Wheels toy car sales.”

To extract this data the Supplier (SUPL) report for Walmart (WMT) is retrieved.

```
data = F.ExtractVectorFormula('WMT','FF_COMPANY_RELATIONSHIP(SUPL,ALL, FEID,ALL,ALL)');
```

Output:

	Security	Relationship	Identifier	Name	Product.Overlap.Number	Product.Overlap.Pct	Revenue.Pct.	Relationship.Direction	Partnership.Type
40	WMT	SUPPLIER	000PK1-E	Mattel, Inc.	0 of 2	0	18.50	Reverse	

This is considered a REVERSE relationship – Mattel has determined this relationship although the search is from WMT .

Mattel also is a supplier to TGT, which accounts for 7.7 % of sales.

```
data = F.ExtractVectorFormula('TGT','FF_COMPANY_RELATIONSHIP(SUPL,ALL, FEID,ALL,ALL)');
```

Output:

	Security	Relationship	Identifier	Name	Product.Overlap.Number	Product.Overlap.Pct	Revenue.Pct.	Relationship.Direction	Partnership.Type
1	TGT	SUPPLIER	000PK1-E	Mattel, Inc.	0 of 2	0	7.70	Reverse	

Customer (CUST) relationships from MAT

From the MATTEL perspective, their relationship to TGT and WMT is as a customer (CUST).

data = F.ExtractVectorFormula('000PK1-E','FF_COMPANY_RELATIONSHIP(CUST,ALL,FEID,ALL,ALL)');

Output:

	Security	Relationship	Identifier	Name	Product.Overlap.Number	Product.Overlap.Pct	Revenue.Pct.	Relationship.Direction	Partnership.Type
1	000PK1-E	CUSTOMER	000YMS-E	Wal-Mart Stores, Inc.	0 of 19	0	18.5	Direct	
2	000PK1-E	CUSTOMER	000XT7-E	Toys R Us Inc.	0 of 1	0	10.8	Direct	
3	000PK1-E	CUSTOMER	002RXT-E	Target Corporation	0 of 2	0	7.7	Direct	

This is considered a DIRECT relationship – Mattel has determined this relationship.

Example 2:

In this example the Partners of Facebook is extracted, the symbol type here is set as Ticker.

data = F.ExtractVectorFormula('FB','FF_COMPANY_RELATIONSHIP(PRTR,ALL, TICKER,ALL,ALL)');

Output:

	Security	Relationship	Identifier	Name	Product.Overlap.Number	Product.Overlap.Pct	Revenue.Pct.	Relationship.Direction	Partnership.Type
1	FB	PARTNERS	MEET-US	MeetMe, Inc.	1 of 1	100	NaN	Reverse	Distribution, Marketing
2	FB	PARTNERS	MMM-US	3M Company	0 of 159	0	NaN	Reverse	Marketing
3	FB	PARTNERS	ATVI-US	Activision Blizzard Inc	0 of 10	0	NaN	Reverse	Marketing
4	FB	PARTNERS	ASR-IT	A.S. Roma S.p.A.	0 of 1	0	NaN	Reverse	Manufacturing
5	FB	PARTNERS	ATA-FR	Atari	0 of 1	0	NaN	Reverse	Distribution
6	FB	PARTNERS	CAR-US	Avis Budget Group, Inc.	0 of 3	0	NaN	Reverse	Marketing

Example 3:

In this example the Private companies that are Competitors to Microsoft are extracted.

data = F.ExtractVectorFormula('MSFT','FF_COMPANY_RELATIONSHIP(COMP,PVT,FEID,ALL,ALL)');

Output:

	Security	Relationship	Identifier	Name	Product.Overlap.Number	Product.Overlap.Pct	Revenue.Pct.	Relationship.Direction	Partnership.Type
1	MSFT		000CXX-E	BMC Software, Inc.		NaN	NaN	Direct	
2	MSFT		00239S-E	Actuate Corporation	5 of 8	62	NaN	Reverse	
3	MSFT		000F2V-E	Compuware Corporation	4 of 10	40	NaN	Reverse	
4	MSFT		062V21-E	Motricity Inc Right	3 of 3	100	NaN	Reverse	

FactSet's Geo Rev Data

Based on the methodology developed by Revere Data, LLC, FactSet Geographic Revenue Exposure provides a normalized and robust display of companies' revenues by geography. With the FactSet Geographic Revenue Exposure database, you can access a comprehensive database that maps disparate geographic revenue of companies to a proprietary four-level classification system containing more than 280 countries, areas, regions, and super-regions.

Geographic revenue represents the percent of geographic revenue derived from customer locations of a company. This data is further accompanied by confidence scores to provide additional insight on the estimates of revenue percentage.

Empowered by a normalized data set mapped to a hierarchical structure, FactSet Revere Geographic Revenue Exposure algorithms have been built to allocate geographic revenues to regions and countries that companies did not explicitly disclose. These estimates are accompanied by a confidence score, which is a ranking score gauging the trustworthiness of the estimates.

Note: FactSet's Geo Rev data requires an additional subscription. Please contact your local support team to gain access.

Formulas:

Super-Regions

+ **FF_GEOREV_SUPER_REGION_PCT**: Returns the percentage of estimated revenue for a super-region.
+ **FF_GEOREV_SUPER_REGION_REV**: Returns the amount of estimated revenue for a super-region. This is calculated as the percentage of estimated revenue for a super-region (**FF_GEOREV_SUPER_REGION_PCT**) multiplied by the FactSet Fundamentals sales value (FF_SALES) for the corresponding fiscal year.
+ **FF_GEOREV_SUPER_REGION_CONF**: Represents the confidence score for revenue percentage for a super-region.

Regions

- **FF_GEOREV_REGION_PCT**: Returns the percentage of estimated revenue for a region.
- **FF_GEOREV_REGION_REV**: Returns the amount of estimated revenue for a region. This is calculated as the percentage of estimated revenue for a region (**FF_GEOREV_REGION_PCT**) multiplied by the FactSet Fundamentals sales value (FF_SALES) for the corresponding fiscal year.
- **FF_GEOREV_REGION_CONF**: Represents the confidence score for revenue percentage for a region.

Areas

- **FF_GEOREV_AREA_PCT**: Returns the percentage of estimated revenue for an area.
- **FF_GEOREV_AREA_REV**: Returns the amount of estimated revenue for an area. This is calculated as the percentage of estimated revenue for an area (**FF_GEOREV_AREA_PCT**) multiplied by the FactSet Fundamentals sales value (FF_SALES) for the corresponding fiscal year.
- **FF_GEOREV_AREA_CONF**: Represents the confidence score for revenue percentage for an area.

Countries

- **FF_GEOREV_COUNTRY_PCT**: Returns the percentage of estimated revenue for a country.
- **FF_GEOREV_COUNTRY_REV**: Returns the amount of estimated revenue for a country. This is calculated as the percentage of estimated revenue for a country (**FF_GEOREV_COUNTRY_PCT**) multiplied by the FactSet Fundamentals sales value (FF_SALES) for the corresponding fiscal year.
- **FF_GEOREV_COUNTRY_CONF**: Represents the confidence score for revenue percentage for a country.

The following table outlines the geographic classification system with region levels, examples, and codes for each:

#	Level	Example	Codes (FactSet Revere Levels, Country ISO)
1	Super-Region	Africa and Middle East, Americas, Asia-Pacific, Europe	R1, R101, R170, R274
2	Region	Latin America, Oceania, European Union	R102, R233, R275
3	Area	Pacific Islands, Caribbean, Western Africa	R237, R103, R45
4	Country	Brazil, United States, Thailand, Germany	76, 840, 764, 276

Formula Syntax:

The following section lists the formula syntax and an example for the _PCT and _CONF formulas available from the FactSet Revere Geographic Revenue Exposure database.

Time Series (FQL) Syntax for _PCT and _CONF Formulas:

- Data = F.ExtractFormulaHistory('Ticker','Code','Date Argument')
  - Where **Code:**
    - ***formula*(*Region Code, Periodicity, Start Date, End Date, Frequency, Alignment***)

Sample for IBM's Geographic Revenue (%) in Africa and the Middle East Super-Region for the last 4 fiscal years

```
+ data = F.ExtractFormulaHistory('IBM-US', 'FF_GEOREV_SUPER_REGION_PCT(\"R1\",ANN,0,-3Y,FY,RF)', '0:-3Y:FY')
```

Time Series (FQL) Syntax for _REV Formulas:

+ Data = F.ExtractFormulaHistory('Ticker','Code','Date Argument')
    + Where **Code:**
        + ***formula*(*Region Code, Periodicity, Start Date, End Date, Frequency, Alignment, Currency***)

Sample for IBM's Geographic Revenue in Africa and the Middle East Super-Region for the last 4 fiscal years.

```
+ data = F.ExtractFormulaHistory('IBM-US', 'FF_GEOREV_SUPER_REGION_REV(\"R1\",ANN,0,-3Y,FY,RF,USD)',
  '0:-3Y:FY')
```

Snapshot (Screening) Syntax for _PCT and _CONF Formulas:

+ Data = F.ExtractFormulaHistory('Ticker','Code','Date Argument')
    + Where **Code:**
        + ***formula*(*Region Code, Periodicity, Start Date, Alignment***)

Sample for IBM's Geographic Revenue (%) in Africa and the Middle East Super-Region for the latest completed fiscal year.

```
+ data = F.ExtractDataSnapshot('IBM-US', 'FF_GEOREV_SUPER_REGION_PCT(R1,ANN,0,RF)', '0Y')
```

Snapshot (Screening) Syntax for _REV Formulas:

+ Data = F.ExtractFormulaHistory('Ticker','Code','Date Argument')
    + Where **Code:**
        + ***formula*(*Region Code, Periodicity, Start Date, Alignment,Currency***)

Sample for IBM's Geographic Revenue in Africa and the Middle East Super-Region for the latest completed fiscal year.

```
+ data = F.ExtractDataSnapshot('IBM-US', 'FF_GEOREV_SUPER_REGION_REV(R1,ANN,0,RF,USD)', '0Y')
```

FactSet GeoRev Use Case for MATLAB/R

Macro to Micro GeoRev Factor to Predict Stock Performance

Full Global Sample – Country GDP Growth	Domestic Firms – Country GDP Growth	Domestic Firms – Alternative Country Factor
16.8% Annual Excess Return	**22.2%** Annual Excess Return	**14.2%** Annual Excess Return
0.93 Sharpe	**0.93** Sharpe	**N/A** Sharpe

AQR	Study Title	Strategy	Universe	Time Horizon	Author	Published
London Business School	Macro to Micro	Expected Country Momentum	Global – 198,315 firm years	12 Years (1998 – 2010)	Richardson, Li & Tuna	Journal of Accounting & Economics 2014

Signal = Incorporating Macroeconomic Information to Firm Level Forecasting

Hypothesis: Stock prices do not efficiently incorporate fundamental information related to country level exposures and expectations of country level performance.

Macrot Calculation with an Example of Intel

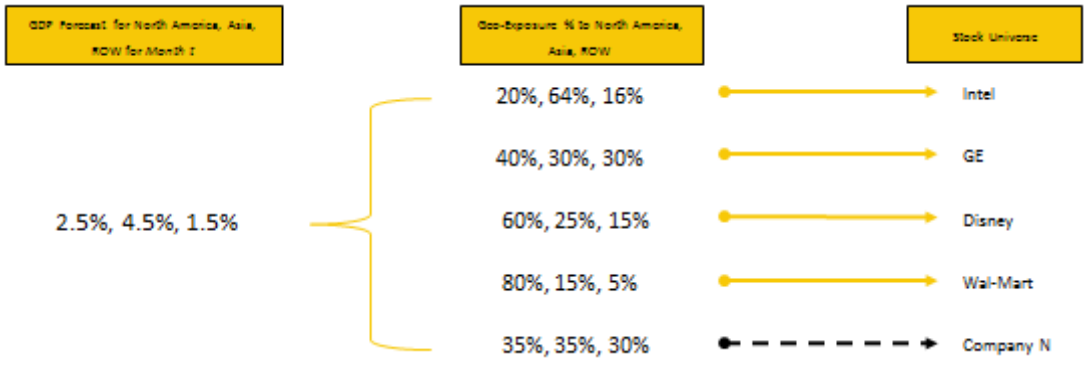
MACROt = Sum Product of firm's geographic revenue exposure and GDP Growth

Geographic Region	Geographic Revenue Exposure (GRE)	Geographic GDP Growth Forecast (GGF)	GRE * GGF
United States	17.6%	3.18%	.0055
China	19.5%	7.01%	.0137
Rest of the World	62.9%	1.5%	.0094

MACROt (Intel) = **.0287**

MACROt Application in Forecasting Returns – Step 1 & 2

1) Start with a global equity universe
2) Identify geographic revenue exposure to regions and GDP growth forecasts for each stock on a monthly basis



MACROt Application in Forecasting Returns – Step 3

3) Calculate Sum Product of GDP growth forecast to geographic revenue exposure, then rank from highest to lowest.

Geo-Exposure % to United States, China, Rest of World	Stock Universe	MACROt Ranking
17.6% | 19.5% | 62.9%	Intel	.0287
47.5% | 6.8% | 45.7%	GE	
68.4% | 3.4% | 28.2%	Disney	
71.1% | 9.8% | 19.0%	Wal-Mart	
35.0% | 35.0% | 30.0%	Company N	.0290

MACROt Application in Forecasting Returns – Step 4 & 5

4) Based on MACROt ranking, break companies into quintile, then long top quintile (cap-weighted) & short bottom quintile (cap-weighted)

5) Repeat steps 2-4 on a monthly basis, adjusting for any changes in GDP growth forecasts and geographic revenue exposure.

R Script

```
# Note: GRE = Geographic Revenue Exposure
# Note: GGF = Geographic GDP Growth Forecast

# Pull Universe Identifiers and Names

CompanyNameRaw = F.ExtractFormulaHistory('', 'PROPER_NAME',
'0','universe','ISON_MSCI_REGION(990100,0,CLOSE,OFF)=1')
IDS = CompanyNameRaw[1]
Date=CompanyNameRaw[2]
CompanyName=CompanyNameRaw[3]

# Pull Geographic GRE Data by Country for the MSCI World Index

USCountryGREPctRaw = F.ExtractFormulaHistory('',
'FF_GEOREV_COUNTRY_PCT(\\"840\\",ANN,0,,,RF)',
'0','universe','ISON_MSCI_REGION(990100,0,CLOSE,OFF)=1')
USCountryGREPct = USCountryGREPctRaw[3]/100
ChinaCountryGREPctRaw = F.ExtractFormulaHistory('',
'FF_GEOREV_COUNTRY_PCT(\\"156\\",ANN,0,,,RF)',
'0','universe','ISON_MSCI_REGION(990100,0,CLOSE,OFF)=1')
ChinaCountryGREPct = ChinaCountryGREPctRaw[3]/100
OtherCountryGREPct = 1-USCountryGREPct-ChinaCountryGREPct

# Pull Geographic GGF Forecast Data by Country for the MSCI World Index
```

```
USCountryGGFRaw =
F.ExtractEconData('','CNS_ECON_DATA(\'RGDPRCY1@US\',0M,0M,M,STEP,AVERAGE,1)')
USCountryGGF = USCountryGGFRaw[3]/100
ChinaCountryGGFRaw =
F.ExtractEconData('','CNS_ECON_DATA(\'RGDPRCY1@CN\',0M,0M,M,STEP,AVERAGE,1)')
ChinaCountryGGF = ChinaCountryGGFRaw[3]/100
OtherCountryGGF = .015

# Calculate GRE * GGF and SUM for MACROt

USGREGGF = USCountryGREPct*USCountryGGF[1,1]
ChinaGREGGF = ChinaCountryGREPct*ChinaCountryGGF[1,1]
OtherGREGGF = OtherCountryGREPct*OtherCountryGGF

SUMGREGGF = USGREGGF+ChinaGREGGF+OtherGREGGF

#Combine All Data and Rename Headers

FullAnalysis =
data.frame(IDS,Date,CompanyName,USCountryGREPct,ChinaCountryGREPct,OtherCountryGREPct,USCoun
tryGGF,ChinaCountryGGF,OtherCountryGGF,USGREGGF,ChinaGREGGF,OtherGREGGF,SUMGREGGF,st
ringsAsFactors = FALSE)

names(FullAnalysis)[1]<-paste("Id")

names(FullAnalysis)[3]<-paste("Company_Name")
names(FullAnalysis)[4]<-paste("United_States_Exposure")
names(FullAnalysis)[5]<-paste("China_Exposure")
names(FullAnalysis)[6]<-paste("Rest_of_World_Exposure")
names(FullAnalysis)[7]<-paste("US_GDP_Forecast")
names(FullAnalysis)[8]<-paste("China_GDP_Forecast")
names(FullAnalysis)[9]<-paste("Rest_of_World_GDP_Forecast")
names(FullAnalysis)[10]<-paste("United_States_GRE_GGF_Calc")
names(FullAnalysis)[11]<-paste("China_GRE_GGF_Calc")
names(FullAnalysis)[12]<-paste("Rest_of_World_GRE_GGF_Calc")
names(FullAnalysis)[13]<-paste("MACROt_Sum_Product")


# Sort Data by MACROt (Sum Product of firm's geographic revenue exposure and GDP growth)

FullAnalysis=FullAnalysis[order(FullAnalysis$MACROt_Sum_Product),]

# Based on MACROt ranking, break companies into quintile, then long top quintile (cap-weighted) &
short bottom quintile (cap-weighted)

QuintileFullAnalysis=quantile(FullAnalysis$MACROt_Sum_Product,probs=seq(0,1,0.20),na.rm=TRUE)

#------------------------------------------------------------------------

i=1
Fractiles=NULL
for (i in i:NROW(FullAnalysis)) {

if(is.nan(FullAnalysis[i,13])) {
Fractile=NaN
} else {
if (FullAnalysis[i,13]>=QuintileFullAnalysis[1] && FullAnalysis[i,13]<QuintileFullAnalysis[2]) {
Fractile=1
} else {
if (FullAnalysis[i,13]>=QuintileFullAnalysis[2] && FullAnalysis[i,13]<QuintileFullAnalysis[3]) {
```

```
Fractile=2
} else {
if (FullAnalysis[i,13]>=QuintileFullAnalysis[3] && FullAnalysis[i,13]<QuintileFullAnalysis[4]) {
Fractile=3
} else {
if (FullAnalysis[i,13]>=QuintileFullAnalysis[4] && FullAnalysis[i,13]<QuintileFullAnalysis[5]) {
Fractile=4
} else {
if (FullAnalysis[i,13]>=QuintileFullAnalysis[5] && FullAnalysis[i,13]<=QuintileFullAnalysis[6]) {
Fractile=5
} else {
Fractile=0
}
}
}
}
}
}
Fractiles=rbind(Fractiles,Fractile)
}

#---------------------------------------------------------------------------

# Create Updated DataFrame

FullAnalysis = cbind(FullAnalysis,Fractiles)


# Upload Data to FactSet for Further Analysis

F.UploadToOFDB('GeoRev_R_FullAnalysis',FullAnalysis)

# Output Data to C: Drive

write.table(FullAnalysis, "c:/users/dsheldon/desktop/GeoRev_UseCase", sep="\t", col.names=NA)

# Create Quintile Portfolios

# Pull Quintile 1

i=1
Quintile1=NULL

for (i in i:NROW(FullAnalysis)) {

if(is.nan(FullAnalysis[i,14])) {
} else {

if (FullAnalysis[i,14]==1){

Quintile1=rbind(Quintile1,FullAnalysis[i,])

} else {


}
}
}

F.UploadToOFDB('GeoRev_R_Sample_Quintile1',Quintile1)
```

```
# Pull Quintile 5

i=1
Quintile5=NULL

for (i in i:NROW(FullAnalysis)) {

if(is.nan(FullAnalysis[i,14])) {
} else {

if (FullAnalysis[i,14]==5){

Quintile5=rbind(Quintile5,FullAnalysis[i,])

} else {


}
}
}


F.UploadToOFDB('GeoRev_R_Sample_Quintile5',Quintile5)
```

MATLAB Script

```
% Note: GRE = Geographic Revenue Exposure
% Note: GGF = Geographic GDP Growth Forecast

% Pull Universe Identifiers and Names
disp('% Pull Universe Identifiers and Names')

CompanyNameRaw = F.ExtractFormulaHistory('', 'PROPER_NAME',
'0','universe','ISON_MSCI_REGION(990100,-1,CLOSE,OFF)=1');
IDs = {CompanyNameRaw.Id};
Date = [CompanyNameRaw.Date];
CompanyName=[CompanyNameRaw.proper_name];

% Pull Geographic GRE Data by Country for the MSCI World Index
disp('% Pull Geographic GRE Data by Country for the MSCI World Index')

USCountryGREPctRaw = F.ExtractFormulaHistory('',
'ZAV(FF_GEOREV_COUNTRY_PCT(\"840\",ANN,0,,,RF)/100)',
'0','universe','ISON_MSCI_REGION(990100,-1,CLOSE,OFF)=1');
USCountryGREPct = [USCountryGREPctRaw.zav];
ChinaCountryGREPctRaw = F.ExtractFormulaHistory('',
'ZAV(FF_GEOREV_COUNTRY_PCT(\"156\",ANN,0,,,RF)/100)',
'0','universe','ISON_MSCI_REGION(990100,-1,CLOSE,OFF)=1');
ChinaCountryGREPct = [ChinaCountryGREPctRaw.zav];
OtherCountryGREPct = 1-USCountryGREPct-ChinaCountryGREPct;
```

```
% Pull Geographic GGF Forecast Data by Country for the MSCI World Index
disp('% Pull Geographic GGF Forecast Data by Country for the MSCI World Index')


USCountryGGFRaw =
F.ExtractEconData('','CNS_ECON_DATA("RGDPRCY1@US",0M,0M,M,STEP,AVERAGE,1)/100');
USCountryGGFPct = USCountryGGFRaw.cns_econ_data;
ChinaCountryGGFRaw =
F.ExtractEconData('','CNS_ECON_DATA("RGDPRCY1@CN",0M,0M,M,STEP,AVERAGE,1)/100');
ChinaCountryGGFPct = ChinaCountryGGFRaw.cns_econ_data;
OtherCountryGGFPct = .015;

USCountryGGF(1:length(IDs))=USCountryGGFPct;
ChinaCountryGGF(1:length(IDs))=ChinaCountryGGFPct;
OtherCountryGGF(1:length(IDs))=OtherCountryGGFPct;

%Calculate GRE * GGF and SUM for MACROt
disp('%Calculate GRE * GGF and SUM for MACROt')

USGREGGF=USCountryGREPct.*USCountryGGF;
ChinaGREGGF=ChinaCountryGREPct.*ChinaCountryGGF;
OtherGREGGF=OtherCountryGREPct.*OtherCountryGGF;

SumGREGGF = USGREGGF+ChinaGREGGF+OtherGREGGF;

% Combine All Data
disp('% Combine All Data ')

FullAnalysis.Id = IDs;
FullAnalysis.Date = Date;
FullAnalysis.Company_Name = CompanyName;
FullAnalysis.UnitedStatesExposure = USCountryGREPct;
FullAnalysis.China_Exposure=ChinaCountryGREPct;
FullAnalysis.Rest_of_World_Exposure=OtherCountryGREPct;
FullAnalysis.US_GDP_Forecast=USCountryGGF;
FullAnalysis.China_GDP_Forecast=ChinaCountryGGF;
FullAnalysis.Rest_of_World_GDP_Forecast=OtherCountryGGF;
FullAnalysis.United_States_GRE_GGF_Calc=USGREGGF;
FullAnalysis.China_GRE_GGF_Calc=ChinaGREGGF;
FullAnalysis.Rest_of_World_GRE_GGF_Calc=OtherGREGGF;
FullAnalysis.MACROt_Sum_Product=SumGREGGF;


% Based on MACROt ranking, break companies into quintile, then long top quintile (cap-weighted) &
short bottom quintile (cap-weighted)
disp('% Based on MACROt ranking, break companies into quintile, then long top quintile (cap-weighted) &
short bottom quintile (cap-weighted)')

Quintile=quantile(FullAnalysis.MACROt_Sum_Product,5);

i=1;
for i = 1:length(FullAnalysis.Id)

   if ((FullAnalysis.MACROt_Sum_Product(i)>=Quintile(1)) &&
((FullAnalysis.MACROt_Sum_Product(i)<Quintile(2))));
      Fractile(i)=1;
   else
   if ((FullAnalysis.MACROt_Sum_Product(i)>=Quintile(2)) &&
((FullAnalysis.MACROt_Sum_Product(i)<Quintile(3))));
```

```
        Fractile(i)=2;
    else
    if ((FullAnalysis.MACROt_Sum_Product(i)>=Quintile(3)) &&
((FullAnalysis.MACROt_Sum_Product(i)<Quintile(4))));
        Fractile(i)=3;
    else
    if ((FullAnalysis.MACROt_Sum_Product(i)>=Quintile(4)) &&
((FullAnalysis.MACROt_Sum_Product(i)<Quintile(5))));
        Fractile(i)=4;
    else
    if ((FullAnalysis.MACROt_Sum_Product(i)>=Quintile(5)));
        Fractile(i)=5;
    else
       Fractile(i)=NaN;
    end
    end
    end
    end
    end

end

FullAnalysis.Quintile=Fractile;
Quintile1 = [];
Quintile5 = [];


% Create Quintile 1 & 5 Portfolios
disp('% Create Quintile 1 & 5 Portfolios')

i=1;

for i = 1:length(FullAnalysis.Id)

    if (isnan(FullAnalysis.Quintile(i)))

continue

    else

    if (FullAnalysis.Quintile(i)==1)

tempQuintile.Id = FullAnalysis.Id(i);
tempQuintile.Date = FullAnalysis.Date(i);
tempQuintile.Company_Name = FullAnalysis.Company_Name(i);
tempQuintile.UnitedStatesExposure = FullAnalysis.UnitedStatesExposure(i);
tempQuintile.China_Exposure=FullAnalysis.China_Exposure(i);
tempQuintile.Rest_of_World_Exposure=FullAnalysis.Rest_of_World_Exposure(i);
tempQuintile.US_GDP_Forecast=FullAnalysis.US_GDP_Forecast(i);
tempQuintile.China_GDP_Forecast=FullAnalysis.China_GDP_Forecast(i);
tempQuintile.Rest_of_World_GDP_Forecast=FullAnalysis.Rest_of_World_GDP_Forecast(i);
tempQuintile.United_States_GRE_GGF_Calc=FullAnalysis.United_States_GRE_GGF_Calc(i);
tempQuintile.China_GRE_GGF_Calc=FullAnalysis.China_GRE_GGF_Calc(i);
tempQuintile.Rest_of_World_GRE_GGF_Calc=FullAnalysis.Rest_of_World_GRE_GGF_Calc(i);
tempQuintile.MACROt_Sum_Product=FullAnalysis.MACROt_Sum_Product(i);
tempQuintile.Quintile=FullAnalysis.Quintile(i);
Quintile1 = [Quintile1 tempQuintile];

        else
```

```
   if (FullAnalysis.Quintile(i)==5)

tempQuintile.Id = FullAnalysis.Id(i);
tempQuintile.Date = FullAnalysis.Date(i);
tempQuintile.Company_Name = FullAnalysis.Company_Name(i);
tempQuintile.UnitedStatesExposure = FullAnalysis.UnitedStatesExposure(i);
tempQuintile.China_Exposure=FullAnalysis.China_Exposure(i);
tempQuintile.Rest_of_World_Exposure=FullAnalysis.Rest_of_World_Exposure(i);
tempQuintile.US_GDP_Forecast=FullAnalysis.US_GDP_Forecast(i);
tempQuintile.China_GDP_Forecast=FullAnalysis.China_GDP_Forecast(i);
tempQuintile.Rest_of_World_GDP_Forecast=FullAnalysis.Rest_of_World_GDP_Forecast(i);
tempQuintile.United_States_GRE_GGF_Calc=FullAnalysis.United_States_GRE_GGF_Calc(i);
tempQuintile.China_GRE_GGF_Calc=FullAnalysis.China_GRE_GGF_Calc(i);
tempQuintile.Rest_of_World_GRE_GGF_Calc=FullAnalysis.Rest_of_World_GRE_GGF_Calc(i);
tempQuintile.MACROt_Sum_Product=FullAnalysis.MACROt_Sum_Product(i);
tempQuintile.Quintile=FullAnalysis.Quintile(i);
Quintile5 = [Quintile5 tempQuintile];

   else

continue

   end
   end
   end
      end


% Calculations Complete
disp('% Calculations Complete')
```

Chapter 4. Integrated Formula Lookup

Sidebar is a formula lookup included in the FactSet workstation that provides a robust way of searching for FactSet request codes along with formula details and definitions. The integration of Sidebar into Statistical packages provides a seamless way of finding FQL and Screening request codes directly from MATLAB and R and also automatically selecting the appropriate factlet.

> **Note:** Minimum FactSet workstation version 2013.A is required to be able to access the SideBar through the Statistical packages.

4.1 Launching Formula Lookup in MATLAB and R

The command prompt to launch the SideBar formula lookup is:

```
F.SideBar()
```

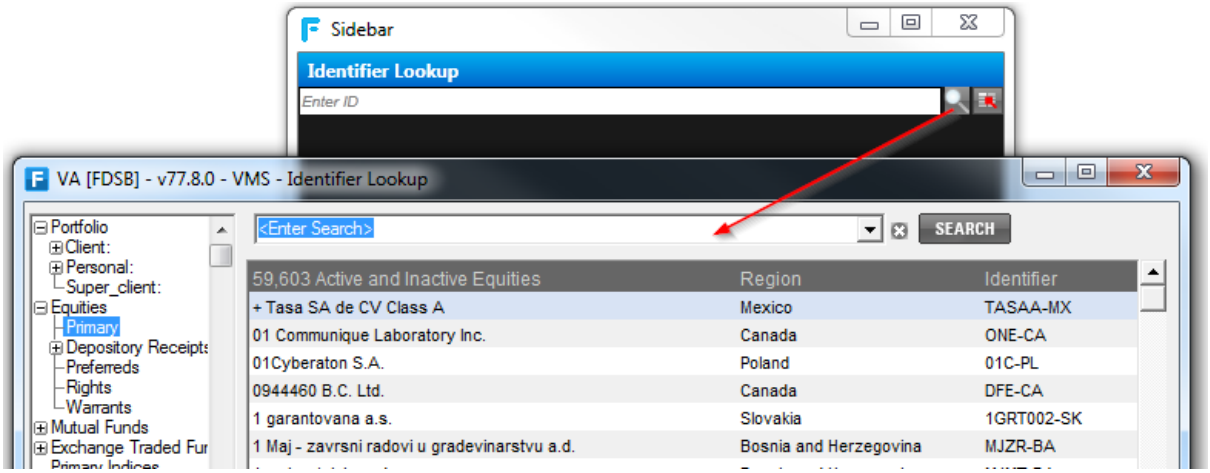
> **Note:** It is necessary to be logged into the FactSet workstation for Sidebar to load. If not already logged in, will be prompted to log into the FactSet workstation when running the F.SideBar() command.

4.2 Searching for Data

The FactSet SideBar allows users to streamline the process of creating FactSet formulas. It features targeted search results, type-ahead search functionality for identifiers and function-building capability.

4.2.1 Identifier Lookup

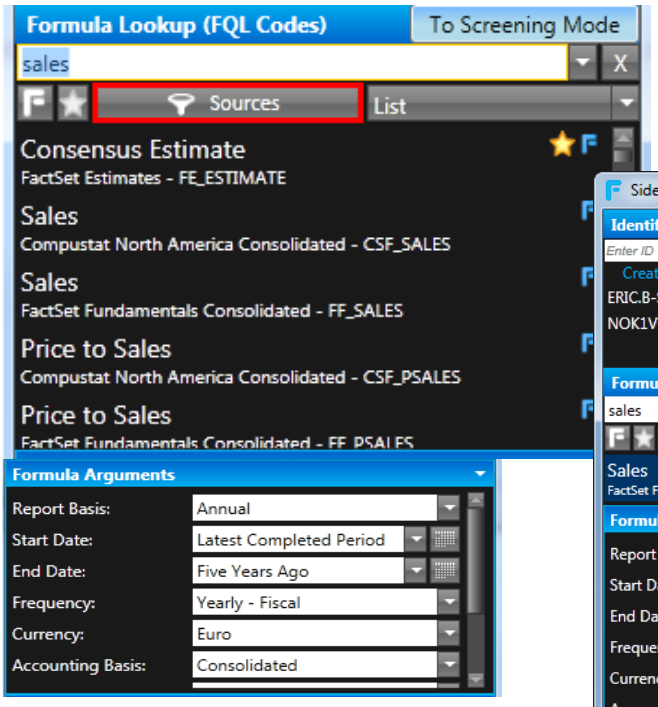
The process of searching for and extracting data in Sidebar involves entering identifiers for equities, indices and bonds, among other categories. The identifier can be entered directly in the Enter ID field which utilizes type-ahead functionality or searched for by launching the Identifier Lookup, as displayed in the screenshot below.



4.2.2 Search for Data Formulas

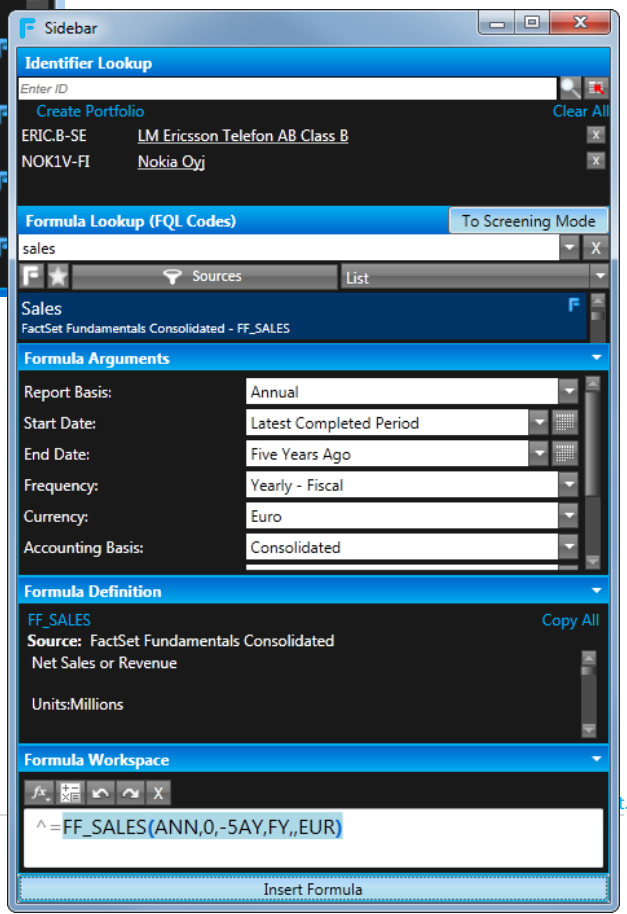
Data Items are searched for by entering keywords in the Formula Lookup section. By default, the returned list of formulas will include matching results from all databases the user has access to use.

To limit the search results it is possible to control which formula libraries are being searched, click the Sources button. To toggle between screening and FQL mode use the button “To Screening/FQL Mode”



Once a formula is selected, a Formula Arguments section loads, in which it is possible to specify the formula specific arguments that are available for the data item selected.

Using Insert Formula, the formula is then inserted into MATLAB or R, translated into the appropriate FactSet function request syntax.



Example:

The SideBar Window to the right displays the input for bringing back Sales data from FactSet Fundamentals for Ericson and Nokia for the last five years. This SideBar input would result in the below FactSet OnDemand syntax to be inserted into MATLAB or R:

```
data = F.ExtractFormulaHistory( 'ERIC.B-SE, NOK1V-FI', 'FF_SALES( ANN,0,-5AY,FY,,EUR)', '0:-5AY:FY')
```

Note: The economic data has not yet been integrated into Sidebar and the Series Lookup in the workstation still needs to be used to search for this data and set up the F.ExtractEconData function. Additionally, multiple identifiers but one data item only can be selected and inserted at one time.

Chapter 5. Troubleshooting & Error Messages

Error Messages

The following are some possible errors when accessing FactSet data in MATLAB.

5.1.1 Undefined Function Error

An error stating that the Function or Class cannot be found usually indicates that the plugin is not properly installed.

- Ensure that the subfolder FactSet3 was created under the root directory of the MATLAB/R version the plugin will be used with.
- For MATLAB only. Ensure that the paths below are included in the MATLAB paths:
  - C:\Program Files\MATLAB\R20xxx\FactSet3

5.1.2. Unauthorized error

The unauthorized error can either be due to

- Not access to the add-on product that is required for OnDemand access[1].

[1] For access related issues please contact your FactSet Sales representative.

- Incorrect OnDemand credentials. The username and password used in OnDemand is different from the FactSet workstation credentials. Please discuss with your FactSet representative if there are any questions.

5.1.3 Proxy settings error

By default the proxy settings are imported from Internet Explorer, if this is not the desired option they can also be entered manually though the advanced settings option in the Configuration Window. Proxy settings are client specific and questions need to be addressed by the end-user's IT department.

5.1.4 Timeout errors

The timeout can be changed through Advanced Options in the Configuration Window. By default the value is 930 seconds but can be adjusted. In general it is more efficient to split queries up and run multiple smaller requests rather than only one large request.

5.1.5 Factlet errors

Errors related to factlet syntax are usually related to either unknown arguments or missing required arguments.

Error	Description
Not enough input arguments.	If any of the required arguments as per Chapter 3 is not defined
Argument named <name of arg> does not have a value.	If a Name - Value pair is not valid
Data downloaded does not contain full header	The data cannot be parsed, check the request code
Types and Column name lines do not match in number of columns.	The data cannot be parsed, check the request code
FactSetQueryError|<error message>	Check the request code

Appendix 1 – Breaking up a request

Script 1: Example of MATLAB Script to Split up a Request

```
% Breakup benchmark into several smaller requests
numChunks = 9; % Edit number of chunks, get CUSIPS
data = F.ExtractBenchmarkDetail('SP50','','');
% Get just the securities
cellArrayOfSecurities = data.SECURITY_ID;
% Figure out how to make a square cell (padding required)
numItemsPerRow = ceil( length(cellArrayOfSecurities)/numChunks );
totalItems = numChunks * numItemsPerRow; % Using ceil to pad
padsNeeded = totalItems - length(cellArrayOfSecurities);
padding = num2cell(nan(1,padsNeeded)); % Make padding matrix
cellArrayOfSecurities = [cellArrayOfSecurities{:} padding]; % Add padding
requestMatrix = reshape( cellArrayOfSecurities, ... % Reshape matrix to rect
                    numChunks, numItemsPerRow );
removeNansFromCellFn = @(x) all(isnan(x(:))); % Declare function clear NaNs
result = [];
for i=1:numChunks,   % Fetch each chunk in order
   cellArrayOfIds = requestMatrix( i,1:end ); % Dont forget to strip NaNs
   cellArrayOfIds(cellfun(removeNansFromCellFn, cellArrayOfIds)) = [];
   currentResult = F.ExtractDataSnapshot(cellArrayOfIds, ...
      'p_name,p_price,p_volume,fg_eps','');
   result = [result currentResult]; %#ok And concatentate results
end
[sx,sx]=sort([result.p_name]); % Sort on names
result=result(sx);           % Arrange
for i = 1:length(result),     % Display
   fprintf('%30s, Price = %6.2f, Volume = %6.2f, EPS = %6.2f\n', ...
   result(i).p_name{:},  result(i).p_price  , ...
    result(i).p_volume  ,     result(i).fg_eps  );
end
```

Script 2: Example of R Script to Split up a Request

```
## Get SP500 benchmark
dfBenchmark = F.ExtractBenchmarkDetail('SP50','','')
## Filter and just get CUSIPS
dfBenchmarkCUSPIPs = dfBenchmark[3]
## Number of CUSIPS
numCUSIPSs = nrow(dfBenchmarkCUSPIPs)
## Decide how many requests to break it up into
numOfRequestChunks = 7
## Create a list of indicies from the benchmark
listOfIndicies = c()
for( i in 1:nrow( dfBenchmarkCUSPIPs ) ){
 listOfIndicies = c( listOfIndicies, dfBenchmarkCUSPIPs[[1]][[i]] );
}
## Split up vectors into chunks - keep even as possible
chunks = split(listOfIndicies,
          rep(1:numOfRequestChunks,
            ceiling(length(listOfIndicies)/numOfRequestChunks),
            length.out = length(listOfIndicies)))
dfResult = NULL; # Declare blank data frame
for( i in 1:length(chunks) ){ # For each chunk and get data
 currentIDsString = paste(chunks[[i]], collapse = ", "); #Build and ID string
 currentDataChunk =
  F.ExtractDataSnapshot( currentIDsString, 'p_name,p_price,p_volume,fg_eps','')
 if( i != 1 ){ ## Then append all the data together into one final frame
  dfResult<-rbind(dfResult, currentDataChunk)
 } else {
  dfResult<-currentDataChunk
 }
}
dfResult = dfResult[order(dfResult$p.name),]
cat('Final result, dfResult, has ', nrow( dfResult ), ' rows and ', ncol(dfResult),' columns.\n',sep='');
```

Appendix 2 – Configuration items

All the settings available from the Configuration Window can also be set programmatically.

The command to set an item is:

F.SetConfigurationItem(F.ConfigOptions.ITEM, 'item')

The Command to get the value for an item is:

F.GetConfigurationItem(F.ConfigOptions.ITEM)

where ITEM is replaced with an item from the below list.

- DataDirectUserName
- DataDirectPassword
- Timeout
- HTTPSRetries
- ProxyUserName
- ProxyPassword
- ProxyServer
- UseManualProxy
- SideBarDataName
- ProxyPort
- UseBeta

Example:

F.SetConfigurationItem(F.ConfigOptions.DataDirectUserName, 'username')
F.GetConfigurationItem(F.ConfigOptions.DataDirectUserName)