import haver — Import data from Haver Analytics databases

Description

Haver Analytics (https://www.haver.com) provides economic and financial databases to which you can purchase access. The `import haver` command allows you to use those databases with Stata. The `import haver` command is provided only with Stata for Windows.

`import haver` *seriesdblist* loads data from one or more Haver databases into Stata's memory.

`import haver` *seriesdblist*`, describe` describes the contents of one or more Haver databases.

If a database is specified without a suffix, then the suffix `.dat` is assumed.

`import haver` accesses Haver Analytics databases that reside on your local network. For accessing Haver Analytics cloud databases, see [D] **import haverdirect**. The two commands employ a near-identical syntax.

Quick start

Describe available time span, frequency of measurement, and source of series E for net fixed assets and consumer durables from the Haver Analytics CAPSTOCK database

```
import haver E@CAPSTOCK, describe
```

Load all available observations for quarterly series ASACX and ASAHS from the US1PLUS database

```
import haver (ASACX ASAHS)@US1PLUS
```

Same as above, but restrict data to the first quarter of 2000 through the fourth quarter of 2010

```
import haver (ASACX ASAHS)@US1PLUS, fin(2000q1,2010q4)
```

Menu

File > Import > Haver Analytics database

Syntax

Load Haver data

```
import haver seriesdblist [ , load_options ]
```

Load Haver data using a dataset of Haver series descriptions stored in memory

```
import haver, frommemory [load_options]
```

Describe contents of Haver database

```
import haver seriesdblist, describe [describe_options]
```

Specify the directory where the Haver databases are stored

```
set haverdir "path" [ , permanently]
```

load_options	Description
`fin(`[*datestring*]`,` [*datestring*]`)`	load data within specified date range
`fwithin(`[*datestring*]`,` [*datestring*]`)`	same as `fin()` but exclude the endpoints of range
`tvar(`*varname*`)`	create time variable *varname*
`case(lower`|`upper)`	read variable names as lowercase or uppercase
`hmissing(`*misval*`)`	record missing values as *misval*
`aggmethod(strict`|`relaxed`|`force)`	set how temporal aggregation calculations deal with missing data
`frommemory`	load data using file in memory
`clear`	clear data in memory before loading Haver database

`frommemory` and `clear` do not appear in the dialog box.

describe_options	Description
* `describe`	describe contents of *seriesdblist*
`detail`	list full-series information table for each series
`saving(`*filename*[`, verbose replace`]`)`	save series information to *filename*`.dta`
`frame(`*framename*[`, verbose replace`]`)`	save series information to *framename*

*`describe` is required.

`collect` is allowed with `import haver`; see **[U] 11.1.10 Prefix commands**.

seriesdblist is one or more of the following:

dbfile

series`@`*dbfile*

`(`*series series* ...`)@`*dbfile*

dbfile is the name of a Haver Analytics database and *series* contains a Haver Analytics series. Wildcards `?` and `*` are allowed in *series*. *series* and *dbfile* are not case sensitive.

Example: `import haver gdp@usecon`
Import series GDP from the USECON database.

Example: `import haver gdp@usecon c1*@ifs`
Import series GDP from the USECON database, and import any series that starts with c1 from the IFS database.

Note: You must specify a path to the database if you did not use the `set haverdir` command.

Example: `import haver gdp@"C:\data\usecon" c1*@"C:\data\ifs"`

If you do not specify a path to the database and you did not `set haverdir`, then `import haver` will look in the current working directory for the database.

Options

Options are presented under the following headings:

Options for import haver

`fin(`[*datestring*]`,` [*datestring*]`)` specifies the date range of the data to be loaded. *datestring* must adhere to the Stata default for the different frequencies. See [D] **Datetime display formats**. Examples are 23mar2012 (daily and weekly), 2000m1 (monthly), 2003q4 (quarterly), and 1998 (annually). `fin(1jan1999, 31dec1999)` would mean from and including 1 January 1999 through 31 December 1999. Note that weekly data must be specified as daily data because Haver-week data are conceptually different from Stata-week data.

`fin()` also determines the aggregation frequency. If you want to retrieve data in a frequency that is lower than the one in which the data are stored, specify the dates in option `fin()` accordingly. For example, to retrieve series that are stored in quarterly frequency into an annual dataset, you can type `fin(1980,2010)`.

If the first *datestring* is not specified, the first date in the series is used as the start of the date range. If the second *datestring* is not specified, the last date in the series is used as the end of the date range.

`fwithin(`[*datestring*]`,` [*datestring*]`)` functions the same as `fin()`, except that the endpoints of the range will be excluded in the loaded data.

`tvar(`*varname*`)` specifies the name of the time variable Stata will create. The default is `tvar(time)`. The `tvar()` variable is the name of the variable that you would use to `tsset` the data after loading, although doing so is unnecessary because `import haver` automatically `tssets` the data for you.

`case(lower` | `upper)` specifies the case of the variable names after import. The default is `case(lower)`.

`hmissing(`*misval*`)` specifies which of Stata's 27 missing values (`.`, `.a`, ..., `.z`) to record when there are missing values in the Haver database.

Two kinds of missing values can be distinguished. The first occurs when Haver has recorded a Haver missing value within the time span covered by a series; by default, these are stored as `.` by Stata, but you can use `hmissing()` to specify that a different extended missing-value code be used. The second occurs when nothing is recorded because the data do not span the entire range; these missing values are always stored as `.` by Stata. The `hmissing()` option does not apply to these observations.

See **[U] 12.2.1 Missing values** for more information on extended missing values.

`aggmethod(strict | relaxed | force)` specifies a method of temporal aggregation in the presence of missing observations. `aggmethod(strict)` is the default aggregation method.

Most Haver series of higher-than-annual frequency have an aggregation type that determines how data can be aggregated. The three aggregation types are average (AVG), sum (SUM), and end of period (EOP). Each aggregation method behaves differently for each aggregation type.

An aggregated span is a time period expressed in the original frequency. The goal is to aggregate the data in an aggregation span to a single observation in the (lower) target frequency. For example, 1973m1–1973m3 is an aggregated span for quarterly aggregation to 1973q1.

`strict` aggregation method:

1) (Average) The aggregated value is the average value if no observation in the aggregated span is missing; otherwise, the aggregated value is missing.
2) (Sum) The aggregated value is the sum if no observation in the aggregated span is missing; otherwise, the aggregated value is missing.
3) (End of period) The aggregated value is the series value in the last period in the aggregated span, be it missing or not.

`relaxed` aggregation method:

1) (Average) The aggregated value is the average value as long as there is one nonmissing data point in the aggregated span; otherwise, the aggregated value is missing.
2) (Sum) The aggregated value is the sum if no observation in the aggregated span is missing; otherwise, the aggregated value is missing.
3) (End of period) The aggregated value is the last available nonmissing data point in the aggregated span; otherwise, the aggregated value is missing. For the last aggregated span of the series, the `strict` aggregation method is applied.

`force` aggregation method:

1) (Average) The aggregated value is the average value as long as there is one nonmissing data point in the aggregated span; otherwise, the aggregated value is missing.
2) (Sum) The aggregated value is the sum if there is at least one nonmissing data point in the aggregated span; otherwise, the aggregated value is missing.
3) (End of period) The aggregated value is the last available nonmissing data point in the aggregated span; otherwise, the aggregated value is missing. This rule is also applied to the last aggregated span of the series.

The following options are available with `import haver` but are not shown in the dialog box:

`frommemory` specifies that each observation of the dataset in memory specify the information for a Haver series to be imported. The dataset in memory must contain variables named `path`, `file`, and `series`. The observations in `path` specify paths to Haver databases, the observations in `file` specify Haver databases, and the observations in `series` specify the series to import.

`clear` clears the data in memory before loading the Haver database.

Options for import haver, describe

`describe` describes the contents of one or more Haver databases.

`detail` specifies that a detailed report of all the information available on the variables be presented.

`saving(`*filename*`[ , verbose replace ])` saves the series meta-information to a Stata dataset. By default, the series meta-information is not displayed to the Results window, but you can use the `verbose` suboption to display it. `replace` specifies that *filename* be overwritten if it exists.

`saving()` saves a Stata dataset that can subsequently be used with the `frommemory` option.

`frame(`*framename*`[ , verbose replace ])` stores the series meta-information to a Stata frame. By default, the series meta-information is not displayed to the Results window, but you can use the `verbose` suboption to display it. `replace` specifies that *framename* be overwritten if it exists.

`frame()` stores a Stata frame that can subsequently be used with the `frommemory` option. You must `frame change` to the specified *framename* before using `import haver` with the `frommemory` option to load the data.

Option for set haverdir

`permanently` specifies that in addition to making the change right now, the `haverdir` setting be remembered and become the default setting when you invoke Stata.

Remarks and examples

Remarks are presented under the following headings:

Installation

Haver Analytics (https://www.haver.com) provides more than 200 economic and financial databases in the form of `.dat` files to which you can purchase access. The `import haver` command provides easy access to those databases from Stata. `import haver` is provided only with Stata for Windows.

Setting the path to Haver databases

If you want to retrieve data from Haver Analytics databases, you must discover the directory in which the databases are stored. This will most likely be a network location. If you do not know the directory, contact your technical support staff or Haver Analytics (https://www.haver.com). Once you have determined the directory location—for example, `H:\haver_files`—you can save it by using the command

```
. set haverdir "H:\haver_files\", permanently
```

Using the `permanently` option will preserve the Haver directory information between Stata sessions. Once the Haver directory is set, you can start retrieving data. For example, if you are subscribing to the USECON database, you can type

```
. import haver gdp@usecon
```

to load the GDP series into Stata. If you did not use `set haverdir`, you would type

```
. import haver gdp@"H:\haver_files\usecon"
```

The directory path passed to `set haverdir` is saved in the `creturn` value `c(haverdir)`. You can view it by typing

```
. display "`c(haverdir)'"
```

Download example Haver databases

There are three example Haver databases you can download to your working directory. Run the `copy` commands below to download HAVERD, HAVERW, and HAVERMQA.

```
. copy https://www.stata.com/haver/HAVERD.DAT haverd.dat
. copy https://www.stata.com/haver/HAVERD.IDX haverd.idx
. copy https://www.stata.com/haver/HAVERW.DAT haverw.dat
. copy https://www.stata.com/haver/HAVERW.IDX haverw.idx
. copy https://www.stata.com/haver/HAVERMQA.DAT havermqa.dat
. copy https://www.stata.com/haver/HAVERMQA.IDX havermqa.idx
```

To use these files, you need to make sure your Haver directory is not set:

```
. set haverdir ""
```

Determining the contents of a Haver database

`import haver` *seriesdblist*`, describe` displays the contents of a Haver database. If no series is specified, then all series are described. Below, we describe the Haver database `haverd.dat`, which we already have on our computer and in our current directory.

```
. import haver haverd, describe
Dataset: haverd
```

Variable	Description	Time span	Frequency	Source
FXTWB	Nominal Broad Trade-W..	03jan2005-02mar2012	Daily	FRB
FXTWM	Nominal Trade-Weighte..	03jan2005-02mar2012	Daily	FRB
FXTWOTP	Nominal Trade-Weighte..	03jan2005-02mar2012	Daily	FRB

```
Summary
    Number of series described: 3
              Series not found: 0
```

By default, each line of the output corresponds to one Haver series. Specifying `detail` displays more information about each series, and specifying *seriesname*`@` allows us to restrict the output to the series that interests us:

```
. import haver FXTWB@haverd, describe detail
FXTWB          Nominal Broad Trade-Weighted Exchange Value of the US$ (1/97=100)
    Frequency: Daily                    Time span: 03jan2005-02mar2012
    Number of observations: 1870        Date modified: 07mar2012 11:27:33
    Aggregation type: AVG               Decimal precision: 4
    Difference type: 0                  Magnitude: 0
    Data type: INDEX                    Group: R03
    Primary geography code: 111         Secondary geography code:
    Source: FRB                         Source description: Federal Reserv..
Summary
    Number of series described: 1
              Series not found: 0
```

You can describe multiple Haver databases with one command:

```
. import haver haverd haverw, describe
  (output omitted)
```

To restrict the output to the series that interest us for each database, you could type

```
. import haver (FXTWB FXTWOTP)@haverd FARVSN@haverw, describe
  (output omitted)
```

Loading a Haver database

`import haver` *seriesdblist* loads Haver databases. If no series is specified, then all series are loaded.

```
. import haver haverd, clear
Summary
─────────────────────────────────────────────────────────────────────────────
       Haver data retrieval: 10 Jul 2024 11:59:42
 Number of series requested: 3
 Number of database(s) used: 1 (HAVERD)
       All series have been successfully retrieved.
Frequency
─────────────────────────────────────────────────────────────────────────────
    Highest Haver frequency: Daily
     Lowest Haver frequency: Daily
 Frequency of Stata dataset: Daily
```

The table produced by `import haver` *seriesdblist* displays a summary of the loaded data and frequency information about the loaded data. For other queries, there may be additional output about query errors and query notes; this is shown only if needed.

The dataset now contains a time variable and three variables retrieved from the HAVERD database:

```
. describe
Contains data
 Observations:         1,870
    Variables:             4
─────────────────────────────────────────────────────────────────────────────
Variable      Storage   Display    Value
    name         type    format    label      Variable label
─────────────────────────────────────────────────────────────────────────────
time            double  %td
fxtwb_haverd    double  %10.0g               Nominal Broad Trade-Weighted
                                               Exchange Value of the US$
                                               (1/97=100)
fxtwm_haverd    double  %10.0g               Nominal Trade-Weighted Exch Value
                                               of US$ vs Major Currencies
                                               (3/73=100)
fxtwotp_haverd  double  %10.0g               Nominal Trade-Weighted Exchange
                                               Value of US$ vs OITP (1/97=100)
─────────────────────────────────────────────────────────────────────────────
Sorted by: time
     Note: Dataset has changed since last saved.
```

Haver databases include the following meta-information about each variable, although the information available will vary depending on the series:

HaverDB	database name
Series	series name
DateTimeMod	date and time the series was last modified
Frequency	frequency of series (from daily to annual) as it is stored in the Haver database
Magnitude	magnitude of the data
DecPrecision	number of decimals to which the variable is recorded
DifType	relevant within Haver software only: if equal to 1, percentage calculations are not allowed
AggType	temporal aggregation type (one of AVG, SUM, or EOP; or, if not defined, one of NA or NA_ANNUAL)
DataType	type of data (for example, ratio, index, US$, or percentage)
Group	Haver series group to which the variable belongs
Geography1	primary geography code
Geography2	secondary geography code (missing if not applicable)
StartDate	start date for data as it is stored in the Haver database
EndDate	end date for data as it is stored in the Haver database
Source	Haver code associated with the source for the data
SourceDescription	description of Haver code associated with the source for the data

When a variable is loaded, this meta-information is stored in variable characteristics (see [P] **char**). Those characteristics can be viewed using `char list`:

```
. char list fxtwb_haverd[]
  fxtwb_haverd[HaverDB]:      HAVERD
  fxtwb_haverd[Series]:       FXTWB
  fxtwb_haverd[DateTimeMod]:  07mar2012 11:27:33
  fxtwb_haverd[Frequency]:    Daily
  fxtwb_haverd[Magnitude]:    0
  fxtwb_haverd[DecPrecision]: 4
  fxtwb_haverd[DifType]:      0
  fxtwb_haverd[AggType]:      AVG
  fxtwb_haverd[DataType]:     INDEX
  fxtwb_haverd[Group]:        R03
  fxtwb_haverd[Geography1]:   111
  fxtwb_haverd[StartDate]:    03jan2005
  fxtwb_haverd[EndDate]:      02mar2012
  fxtwb_haverd[Source]:       FRB
  fxtwb_haverd[SourceDescription]:
                              Federal Reserve Board
```

You can load multiple Haver databases/series with one command. To load the series FXTWB and FXTWOTP from the HAVERD database and all series that start with V from the HAVERMQA database, you would type

```
. import haver (FXTWB FXTWOTP)@haverd V*@havermqa, clear
  (output omitted)
```

`import haver` automatically `tssets` the data for you. You can issue `tsset` to see how the data are currently set.

Loading a Haver database from a describe file

You often need to search through the series information of a Haver database or databases to see which series you would like to load. You can do this by saving the output of `import haver, describe` to a Stata dataset with the `saving(`*filename*`)` option. The dataset created can be used by `import haver, frommemory` to load data from the described Haver databases. For example, here we search through the series information of database HAVERMQA.

```
. import haver havermqa, describe saving(my_desc_file)
  (output omitted)
. use my_desc_file, clear
. describe
Contains data from my_desc_file.dta
 Observations:           161
    Variables:             8                  10 Jul 2024 11:59
```

Variable name	Storage type	Display format	Value label	Variable label
path	str1	%-9s		Path to Haver file
file	str8	%-9s		Haver filename
series	str7	%-9s		Series name
description	str80	%-80s		Series description
startdate	str7	%-9s		Start date
enddate	str7	%-9s		End date
frequency	str9	%-9s		Frequency
source	str3	%-9s		Source

```
Sorted by:
```

The resulting dataset contains information on the 164 series in HAVERMQA. Suppose that we want to retrieve all monthly series whose description includes the word "Yield". We need to keep only the observations from our dataset where the frequency variable equals "Monthly" and where the description variable contains "Yield".

```
. keep if frequency=="Monthly" & strpos(description,"Yield")
(152 observations deleted)
```

To load the selected series into Stata, we type

```
. import haver, frommemory clear
```

Note: We must `clear` the described data in memory to load the selected series. If you do not want to lose the changes you made to the description dataset, you must save it before using `import haver, frommemory`.

The `frame(`*framename*`)` option works similarly to the `saving(`*filename*`)` option, but instead of saving a file to disk, `frame()` stores the metadata in a frame. See [D] **frames** for more information on data frames.

Temporal aggregation

If you request series with different frequencies, the higher-frequency data will be aggregated to the lowest frequency. For example, if you request a monthly and a quarterly series, the monthly series will be aggregated. In rare cases, a series cannot be aggregated to a lower frequency and so will not be retrieved. A list of these series will be stored in `r(noaggtype)`.

The options `fin()` and `fwithin()` are useful for aggregating series by hand.

Daily and weekly data

Daily and weekly queries require additional explanations because these frequencies are implemented differently in Haver databases than in Stata datasets. A Haver daily series refers to a business daily frequency, which is five days per week and counts only Monday through Friday as observations. An exact match for Haver daily is Stata's business daily frequency (`%tb`), which uses a business-day calendar that excludes weekends and includes all weekdays throughout the year. Stata's daily frequency (`%td`), by contrast, counts all seven days of a week. This frequency is called 7-daily in Haver databases.

The implementations of the weekly frequency also differ between Haver and Stata. Haver's implementation counts one week after another, without any reference to the calendar year, thereby allowing for years that mostly have 52 observations but sometimes have 53 observations. Each Haver weekly data series has a value set for its controlling-day-of-week (CDOW) property. This is typically the weekday on which the data are released by the source. This information is preserved in a Stata characteristic called `cdow`. For example, the `cdow` characteristic for series `SP100@WEEKLY` is Friday.

Stata's `%tw` frequency counts weeks from the beginning of the year and caps the week number at 52. There are two ways in which Haver's weekly frequency can be exactly matched in Stata: either in Stata's daily frequency (`%td`) in combination with a delta of seven days (see [TS] **tsset**) or in a datetime business calendar (`%tb`), which here should count only one particular weekday as a valid business day.

The above discrepancies in frequency implementations are resolved in `import haver` in the following way: any query that exclusively consists of one or more of Haver 7-daily, Haver daily, or Haver weekly series results in a Stata dataset of Stata daily frequencies (`%td`). In that dataset,

> Haver 7-daily series receive rows for all seven days of the week covered by their span.
>
> Haver daily series receive rows for five days of the week (Monday through Friday) covered by their span.

Haver weekly series are assigned dates that correspond to their CDOW. For example, series `SP100@WEEKLY` has a CDOW of Friday and, at the time of writing, covers the time span 06jan1989–31may2024. Thus, in the Stata dataset, `SP100@WEEKLY` receives rows with dates 06jan1989, 13jan1989, …, 24may2024, 31may2024 (these are all Fridays).

Note that if a query combines Haver 7-daily series and Haver daily series with one or more Haver weekly series, aggregation to weekly values is performed. For Haver 7-daily series, the values Monday through Sunday are aggregated to a single value, and the resulting (Haver weekly) series receives a CDOW of Sunday, with corresponding (Sunday) rows in the dataset. Similarly, for Haver daily series, the values Monday through Friday are aggregated to a single value, and the resulting (Haver weekly) series receives a CDOW of Friday, with corresponding (Friday) rows in the dataset.

Once you have queried your daily and weekly series, you may want to use Stata's `tsset` or business calendar features to further tailor the dataset toward your needs. Several ways for you to do this were indicated above.

When you aggregate Haver daily and Haver weekly series to lower frequencies (for example, monthly), the above considerations are not relevant. Another issue with these data is Haver aggregation modes. Haver daily and Haver weekly series often contain missing values due to events such as national holidays. When you aggregate to a lower frequency under the default aggregation mode `strict`, such missing values then result in aggregated values that are also set to missing. In such cases, you may want to consider using aggregation modes `relaxed` or `force`. See option `aggmethod()` for more details.

Stored results

`import haver` stores the following in `r()`:

Scalars

`r(k_requested)`	number of series requested
`r(k_noaggtype)`	number of series dropped because of invalid aggregation type
`r(k_nodisagg)`	number of series dropped because their frequency is lower than that of the output dataset
`r(k_notinrange)`	number of series dropped because data were out of the date range specified in `fwithin()` or `fin()`
`r(k_notfound)`	number of series not found in the database

Macros

`r(dbnamelist)`	list of Haver databases used in command
`r(noaggtype)`	list of series dropped because of invalid aggregation type
`r(nodisagg)`	list of series dropped because their frequency is lower than that of the output dataset
`r(notinrange)`	list of series dropped because data were out of the date range specified in `fwithin()` or `fin()`
`r(notfound)`	list of series not found in the database

`import haver, describe` stores the following in `r()`:

Scalars

`r(k_described)`	number of series described
`r(k_notfound)`	number of series not found in the database

Macros

`r(notfound)`	list of series not found in the database

Also see

[D] **import** — Overview of importing data into Stata

[D] **import delimited** — Import and export delimited text data

[D] **import fred** — Import data from Federal Reserve Economic Data

[D] **import haverdirect** — Import data from Haver Analytics cloud servers

[D] **jdbc** — Load, write, or view data from a database with a Java API

[D] **odbc** — Load, write, or view data from ODBC sources

[TS] **tsset** — Declare data to be time-series data