User Manual

Version 16

TOC

Single User License Agreement and Limited Warranty for Stat/Transfer

STAT/TRANSFER
SINGLE-USER LICENSE AGREEMENT
AND LIMITED WARRANTY

– IMPORTANT –
READ CAREFULLY BEFORE INSTALLING STAT/TRANSFER SOFTWARE.

By installing or using any portion of Stat/Transfer Software (the "**SOFTWARE**"), you accept all of the following terms and conditions of this **Stat/Transfer Single-User License Agreement and Limited Warranty** (the "LICENSE AGREEMENT").

LEGAL EFFECTIVENESS

THIS IS A LEGAL AGREEMENT BETWEEN CIRCLE SYSTEMS, INC., A WASHINGTON, USA, CORPORATION, THE COPYRIGHT OWNER OF THE STAT/TRANSFER ("**WE**") AND YOU, AS AN INDIVIDUAL AND NOT AN ENTITY, THE PURCHASER OF A LICENSE TO THE STAT/TRANSFER SOFTWARE AND/OR THE INDIVIDUAL PERSON (NOT AN ENTITY) WHO IS THE END USER OF THE SOFTWARE **("YOU**").

CAREFULLY READ THIS LICENSE AGREEMENT BEFORE INSTALLING, USING, OR OPENING A PACKAGED COPY OF THE STAT/TRANSFER SOFTWARE. **WE WILL NOT ACCEPT ANY PURCHASE ORDER OR SELL YOU A LICENSE TO INSTALL AND USE THE SOFTWARE UNLESS YOU AGREE TO <u>ALL</u> OF THE TERMS OF THIS LICENSE AGREEMENT.** IF YOU DO NOT AGREE TO THESE TERMS, DO NOT OPEN, INSTALL, OR USE THE SOFTWARE ON YOUR COMPUTER OR OPEN A PACKAGED COPY. REMOVE ANY AND ALL COPIES OF THE SOFTWARE FROM YOUR COMPUTER AND RETURN THE SOFTWARE AND ANY ACCOMPANYING MATERIALS WITHIN THIRTY (30) DAYS OF PURCHASE, WITH PROOF OF PURCHASE, FOR A FULL REFUND OF THE AMOUNT YOU ORIGINALLY PAID FOR THE SOFTWARE.

SUMMARY OF YOUR LICENSE

We offer single-user licenses, single-user and volume single-user license annual sub-scriptions, and annual network licenses (lease). The terms and conditions of this LICENSE AGREEMENT apply to all licenses.

Single-user licenses allow YOU to install Stat/Transfer on up to three machines on the local drive, as long as YOU are the primary user of those machines and the only user of Stat/Transfer. For instance, YOU can install Stat/Transfer on your office computer, your home computer, and your laptop. Single user licenses will not work over a network and they should not be used under Terminal Server, Citrix, or in similar environments.

.

Network licenses allow for the installation over a network for a set quantity of users. Every individual who accesses Stat/Transfer during the license term at least once must be covered under a license. We do not offer concurrent, floating, or simultaneous use or any otherwise sharing of each single license.

SOFTWARE LICENSE

This LICENSE AGREEMENT grants YOU a license to the SOFTWARE. YOU are a licensee and not an owner of copies of the SOFTWARE. This LICENSE AGREEMENT, as well as copyright and other intellectual property laws and inter-national treaties protect this SOFTWARE, and set forth your specific rights to use the SOFTWARE. The provisions of this LICENSE AGREEMENT are conditions of YOUR license. If you materially violate a condition, then YOUR use of the SOFTWARE is without authority and may therefore make YOU liable for infringing OUR copyright in the SOFTWARE.

If you have a single user license WE grant YOU the right to install the SOFTWARE on up to three computers (each a "**DEDICATED COMPUTER**"), as long as YOU are the primary user of those computers, the only user of the SOFTWARE, and to use the SOFTWARE on a single one of those computers at a time. A DEDICATED COMPUTER is one that you use at least 80% of the time that it is in use,

YOU may make a single additional copy of the SOFTWARE for backup or archival pur-poses. If YOU receive a copy of the SOFTWARE electronically and on disk, YOU may use the disk copy for archival purposes only. Copyright law prohibits YOU from making any other copy of the SOFTWARE and user manual without OUR permission.

For perpetual licenses, YOUR license to the SOFTWARE is perpetual as long as YOU observe the conditions of this LICENSE AGREEMENT. For annual licenses, YOUR

license to the SOFTWARE is valid for one year and as long as YOU observe the conditions of this LICENSE AGREEMENT.

YOU may not alter, modify, or adapt the SOFTWARE or user manual, or create any derivative works based on them. WE distribute the SOFTWARE in computer executable form only, and do not allow user access to the underlying source code and data. YOU may not reverse engineer, decompile, or disassemble the SOFTWARE to gain access to such code and data, except to the extent applicable law expressly permits such activity. Decompiling or disassembling the SOFTWARE may also violate the SOFTWARE's copyright.

YOU may not use the SOFTWARE to provide data conversion services for others, whether inside or outside your organization. A license to use the SOFTWARE is necessary to utilize the SOFTWARE's data conversion functionality.

A single user license does not allow the user of the SOFTWARE as a component in a software as service arrangement, a component in a web server or service or in a production environment or in a production, or batch environment in which the same set of commands are run more than five times in an annual period. Specialized licenses are available on request for use cases such as these. We do not offer concurrent, floating, or simultaneous use or any otherwise sharing of each single license.

YOU may not sublicense, sell, rent, lend, lease, sublicense, or give away the SOFTWARE to others. YOU may, however, with OUR prior written permission, transfer the SOFTWARE, written materials, and this LICENSE AGREEMENT as a package if the other party registers with US and agrees to accept this LICENSE AGREEMENT. This provision is intended to allow users within businesses and government to transfer their license to the SOFTWARE in a job reassignment. Granting permission, however, is within OUR sole discretion, YOU may not transfer a license originally sold in a volume or network license unless YOU transfer all the licenses at the site. YOU may not retain any copies of the SOFTWARE yourself once YOU have transferred it.

Any unauthorized copying, distribution, or modification of the SOFTWARE will automatically cancel your license to use the SOFTWARE and violate the SOFTWARE's copyright.

SOFTWARE SUPPORT

This LICENSE AGREEMENT entitles YOU to technical support during the term of your subscription or lease, or for perpetual license, until two major versions after yours have shipped.After those periods, WE offer paid technical support subscriptions to licensees.

WE will provide reasonable technical support to answer questions and address problems concerning the operation and use of the SOFTWARE. Technical support is provided in English. Technical support is available exclusively by use of our a form on our website. This LICENSE AGREEMENT also entitles YOU to receive minor version updates to the SOFTWARE that WE publicly release. WE provide these update releases only covering up to two versions back from the current version of the SOFTWARE. Licensees requiring updating covering more than two version releases must purchase a new license to the SOFTWARE.

LIMITED WARRANTY AND REMEDIES

WE warrant that the SOFTWARE will perform in substantial compliance with the specifications set forth in the user manual provided with the SOFTWARE, provided that it is not modified and it is used on computer hardware and with an operating system for which it was designed. In order to qualify for a claim of defective SOFTWARE and requesting a refund, YOU must have worked with a technician to remedy the problem first via our helpdesk and deactivated the software from any and all computers.

These warranties are limited to the 90-day period from YOUR original purchase of a license to the SOFTWARE. If YOU report in writing within 90 days of purchase a substantial defect in the SOFTWARE's performance, WE will attempt to correct it or, at OUR option, authorize a refund of the amount YOU originally paid for YOUR license to the SOFTWARE.

THE WARRANTIES AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN PLACE OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED, CONCERNING THE SOFTWARE OR USER MANUAL. WE DO NOT AND CANNOT WARRANT THE PERFORMANCE OR RESULTS YOU MAY OBTAIN USING THE SOFTWARE. EXCEPT FOR THE FOREGOING LIMITED WARRANTY AND REMEDIES, AND FOR ANY WARRANTY, CONDITION, REPRESENTATION, OR TERM TO THE EXTENT TO WHICH IT CANNOT OR MAY NOT BE EXCLUDED OR LIMITED BY LAW APPLICABLE TO YOU IN YOUR JURISDICTION. WE MAKE NO OTHER WARRANTIES, REPRESENTATIONS, OR CONDITIONS, EXPRESS OR IMPLIED, WITH RESPECT TO THE SOFTWARE, MEDIA, OR USER MANUAL, INCLUDING THEIR MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OF THIRD PARTY RIGHTS, INTEGRATION, OR FITNESS FOR A PARTICULAR PURPOSE.

WE WILL IN NO EVENT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF OR INABILITY TO USE THE SOFTWARE OR USER MANUAL, EVEN IF WE HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Because software in general is inherently complex and may not be completely free of errors, WE are not responsible for any costs including, but not limited to, lost profits or revenue, loss of time or use of the SOFTWARE, loss of data, the cost of recovering software or data, the cost of substitute software, claims by third parties, or similar costs. In no event will the liability of Circle Systems exceed the amount paid for the license to the SOFTWARE.

NOTICES TO GOVERNMENT LICENSEES

The SOFTWARE and manual are "Commercial Items," as that term is defined at 48 CFR 2.101, consisting of "Commercial Computer Software" and "Commercial Computer Software Documentation," as such terms are used in 48 CFR 12.212 or 48 CFR 27.7202, as applicable. Consistent with 48 CFR 12.212 or 48 CFR 227.7202-1 through 227.7202-4, as applicable, the Commercial Computer Software and Commercial Computer Software Documentation are being licensed to U.S. Government end users only as Commercial Items and with only those rights as are granted to all other end users pursuant to the terms and conditions of this LICENSE AGREEMENT. Unpublished-rights reserved under the copyright laws of the United States. Circle Systems, Inc., 1001 Fourth Avenue, Suite 3200, Seattle, WA 98154 USA.

Any State, instrumentality of a State, and officer or employee of a State or its instrumentality acting in their official capacity will not be immune under any doctrine of sovereign immunity from suit for violating this LICENSE AGREEMENT or the SOFTWARE's copyright.

COMPLIANCE

WE may conduct a reasonable audit of your compliance with this LICENSE AGREEMENT and will be given access to the machines on which the SOFTWARE is installed for purposes of the audit. The audit will be initially conducted at OUR expense, but YOU will be responsible for the reasonable cost of the audit if it discloses a material violation of this LICENSE AGREEMENT or the SOFTWARE's copyright.

GENERAL

The laws of the State of Washington, USA govern this LICENSE AGREEMENT. Any lawsuit to enforce this LICENSE AGREEMENT or the SOFTWARE's copyright may be brought and tried in the state or federal courts within King County, Washington, USA. The substantially prevailing party in any lawsuit will be entitled to recover its reasonable attorney's and expert's fees and expenses.

This LICENSE AGREEMENT is the complete and exclusive statement of the agreement between YOU and US concerning use of the SOFTWARE. It supersedes any prior agreement or understanding, oral or written, between YOU and US, or our respective agents

and employees, with respect to this subject. None of OUR distributors, resellers, dealers, or agents is authorized to make any modification, extension, or addition to this LICENSE AGREEMENT and the limited warranty and limitation of liability included in this LICENSE AGREEMENT.

Introduction

What Stat/Transfer Does

Stat/Transfer is designed to simplify the transfer of statistical data between different programs.

Data generated by one program is often needed in another context either for analysis, for cleaning and correction, or for presentation. However, not only must the data be transferred, but the variables generally must be re-described for each program with additional information, such as variable names, missing values and value and variable labels. This process is not only time-consuming, it is error-prone. For those in possession of data sets with many variables, it represents a serious impediment to the use of more than one program.

Stat/Transfer removes this barrier by providing an extremely fast, reliable, and automatic way to move data.

Stat/Transfer will automatically read statistical data in the internal format of one program and will transfer as much of the information as is present and appropriate to the internal format of another.

Stat/Transfer preserves all of the precision in your data by storing it internally in double precision format. However, on output, it will, where possible, automatically minimize the size of your output data set by intelligently choosing data storage types that are only as large as necessary to preserve the input precision. Stat/Transfer also allows precise and easy manual control over the storage format of your output variables, in case this is necessary.

In addition to converting the formats of variables, Stat/Transfer also processes missing values automatically.

Stat/Transfer can save hours, even days of manual labor while, at the same time, eliminating error. Furthermore, you gain this speed and accuracy without losing flexibility, since Stat/Transfer allows you to select only the variables and cases you want to transfer.

In addition to the standard graphical user interface, a command processor allows you to run a transfer in batch mode, using a command file. The user interface can automatically generate a command file to exactly reproduce your data transfer operations. This makes it straightforward to set up fully automatic batch procedures for repetitive tasks and allows you to precisely document the work that you have done.

Supported File Types

Version 16 of Stat/Transfer will support the file types given below:

1-2-3	Minitab
Access	NLOGIT
ASCII - Delimited	ODBC
ASCII - Fixed Format	OpenDocument Spreadsheets
BayesiaLab (Write Only)	OSIRIS (read-only)
dBASE and compatible formats	Paradox
	Quattro Pro
DDI Schemas	R
Epi Info	RATS
Excel	SAS Data
Feather	SAS CPORT (read only)
FoxPro	SAS Program Files + ASCII
Gauss	SAS Transport
gretl	S-PLUS
HTML tables	SPSS Data
JMP	SPSS Program + ASCII
JSON-Stat (Read Only)	SPSS Portable
LIMDEP	Stata
Matlab	Statistica
Mineset	SYSTAT
	Triple-S

In addition, for data archiving and exchange, Stat/Transfer will write ASCII data together with programs to read the data back into SAS, SPSS, and Stata. It also supports its own, easy to use, Schema format for reading in external data in ASCII format.

See the topics for specific file types for more information on supported versions of each type.

What's New in Stat/Transfer

New Formats

Stat/Transfer version 16 has added support for the following formats:

- Reading SPSS syntax files and ASCII data
- Reading SAS program files and ASCII data
- Reading and Writing Minitab 19 project and worksheet files
- Reading R single object (.rds) files
- Reading and Writing Apache Big Data formats through ODBC:
  - Drill
  - Hive
  - Impala

New Features

- Very easy-to-use, Stata-style date and time input masks for reading ASCII data
- Localized month names for reading non-English dates
- Improved support for extended missing values
- Increased user control over dataset optimization
- Increased user control over the handling of newlines in ASCII data

New Platform Support

Stat/Transfer is now shipping with native support for Mac M1 processors and full support for Big Sur and Monterey. Windows 11 is fully supported.

Getting Started

Trial Mode Software

If you have downloaded Stat/Transfer from our website, or otherwise obtained a trial version, it will work as if it were a full version, except that one out of approximately sixteen cases will be deleted from your output file. A message box will warn you of this, and the border at the top of the Stat/Transfer window will say "Trial Mode".

To turn the trial version into a fully functioning copy of the software, go to our website and purchase a copy with the appropriate license. As soon as the order is processed, you will be sent an activation code by email.

Activation

After installation is complete, you must activate your software with a code or license file that was emailed to you. It is easiest if you have an internet connection during the activation process.

Activation
Online
First, go the **About** tab and press the **Activate Online** button. On the next screen, enter your activation code. Then press **Next**. You will be asked to enter your name, your organization and your email address. After you press **Next** again, you will be asked to enter a password, which will be used if you reactivate your software on another computer (see below). Because it may be emailed to you in plain text, you should not use a valuable password.

You will also be asked to choose a *computer name*. This is a name that you make up, and is only used within Stat/Transfer. You should choose a computer name that will clearly identify your computer, such as "laptop" or "office." This name must be under 32 characters. Each of your computers must have a unique name. Once the computer is activated, you will not be able to change the name.

Finally, when you press **Next** again, your information will be sent to our server and, if your serial number is valid, the activation information will be written to your computer. Once activation is complete, you must restart Stat/Transfer.

Offline
If your computer is not connected to the internet, you will be offered an activation process by which the program will generate a small file that you can save onto a thumb drive or other portable media. You can then take this to an internet-connected computer and upload it to our website. The server will return another file that you can then save to disk. Then, at your original computer, you can activate your copy by clicking the **Install License** button on the **About** tab and then navigating to the file you have downloaded and opening it. It Stat/Transfer will then save the file on your disk and activate your copy.

Help in Activating the Software
For detailed instructions, please see the **Support/Activation** section of our website, www.stattransfer.com.

Installing on Additional Computers

The activation process will send a "machine fingerprint" that identifies your particular computer. If you have a single-user license for Stat/Transfer, you are permitted to install it on up to three computers, as long as you are the primary user of the computers (for instance on a home computer, office computer, and a laptop). If you get a new computer, you can deactivate one of the machines and activate on the new computer.

In order to install Stat/Transfer on another machine using the same activation code, you will need to re-activate the software after you install the program on the second machine. The re-activation process will ask you for your old password and asks you to create a new password that identifies your second installation. If you have forgotten or mis-placed your old password, you can have it sent to the email address you gave in your ini-tial activation session by clicking on the **Forgot your Password?** button. Remember to write down both your old and new passwords.

Deactivation

About Deactivation

Your copy of Stat/Transfer allows you to install Stat/Transfer on three separate computers. Typically these will be a work computer, a home computer and a laptop.

If you have used all three, you must deactivate one before activating on a fourth.

To Deactivate Stat/Transfer:

1) On the "About" tab, click the **Deactivate** button.

2) A dialogue box will give an "Are you sure?" message. Click **Yes**.

3) When the program confirms you are activated, close the program and proceed to activate on your new machine

Remote Deactivation

If you no longer have access to the computer that you wish to deactivate, you can deactivate on a website. Go to:

https://stattransfer.com/offline_deact.html

You will need to know your activation code, your computer name, and your password.

Network and Lease Licensing

Licensing

A lease license allows a fixed number of users to use Stat/Transfer within a year.

If you buy a lease license for Stat/Transfer, activation codes are not used. Instead, you will receive a small text file that contains your licensing information.

As long as the total number of users is below the maximum allowed by the license, you can install Stat/Transfer on any combination of computers onany operating system (Windows, Mac, or Linux). You can also install Stat/Transfer for shared use on a network server or terminal server environment (such as Citrix).

In most cases, the license is installed by clicking on the **Install License** button on the **About** tab. This brings up a file-open dialog, which you can use to select the license file. At that point, you can select whether you want to install for network or single computer use.

Special Instructions for Installing Lease Licenses for Windows Users

For network use, the license must be stored in the same location as the executable. However, Windows will not allow programs to write to directories under the Program Files directory.

Therefore, if you are going to install Stat/Transfer for use on a network, you must install the program in some other directory than the default Program Files location. This location should be visible and permitted for use by your network users.

If you are installing for use on a single computer, you can install Stat/Transfer anywhere, including under Program Files and the license will be installed in an Application Data subdirectory.

The Redirect License

Because licenses must be renewed annually, this can create a secondary problem of updating the licenses on many distributed computers. To work around this problem, you can install a redirect license on each computer that points to an actual license which is stored in a location of your choice on the network.

The redirect license is called *license.dat*. It should have the following form:

```
[Redirect]
  Path=network_path/license.dat
```

Where *network_path* is the location of the actual license.dat file.

Installing the Redirect License

The redirect file can be installed by selecting a copy of the redirect license on your disc after pressing the install license button, or it can be installed by the installation program in batch mode as described below.

Installation Options

Several options are available for installing Stat/Transfer in a network environment. The installer is named *stdemo.exe*.

Command Line Switches

The following command line switches are available for the Windows installer:

* /S (must be capitalized) installs silently without prompting
* /license copies a license (or redirection file) from the setup directory to the installation directory
* /license-sc copies the license (or redirection file) to the Program Data directory for Stat/Transfer
* /D=<install_dir> installs the program in install_dir

Note that the /D= switch must be the last parameter in the command line and must not contain any quotes, even if the path contains spaces. Only absolute paths are supported.

Obviously, the /license and /license-sc switches are mutually exclusive and cannot be used at the same time. The /license switch will copy the license to anywhere the program is installed because, under Windows, installation programs have privileges that other programs do not. For both of these options, the license or redirect file must be in the same directory as the installation program and must be named **license.dat**.

Read-me File

The installation or Web update procedure may copy a file called *read.me*, which will be a supplement to the on-line help or manual. There is a shortcut to the *read.me* file from the Windows **Start** menu, in the Stat/Transfer group.

You should check to see if the *read.me* file exists or has been updated. If so, you can read it in any editor or word-processor.

We make every effort to keep up with changes in the file formats of popular software and the *read.me* file will contain the latest information on which versions of these programs are supported. The file will also contain the latest information on other improvements to Stat/Transfer.

You can also get current information about Stat/Transfer by visiting our website at **https://stattransfer.com**.

You can reach our website from the Stat/Transfer **About** screen.

Demo Files

The distribution disk contains sample files in many of the supported formats, which you may find useful in learning about Stat/Transfer's capabilities.

The file name indicates which program format each file illustrates. In addition there is a file, *demo.sav*, that illustrates the way Stat/Transfer treats different kinds of variables. The installation program will copy these files to the same directory chosen for installation of Stat/Transfer.

Web Update

We periodically post maintenance releases of Stat/Transfer on our website to support new file formats, add features, or to fix problems that have come to our attention. However, we have found that many people are not taking advantage of these releases, so they are using software that is older than it should be. To address this problem, Stat/Transfer will automatically check the Web for updates.

By default, the program will check for new versions once every week, but you can change this option. You can check immediately, daily, weekly, monthly, quarterly, or (if you are running on a computer that is not connected to the Web) never.

In order to change the interval at which Stat/Transfer checks the Web, click on the **About** tab of Stat/Transfer and select one of the options.

Suppose you choose **Every Week** (the default). Each time you start Stat/Transfer, the program will compare the current date to the date at which the version was last checked. If it the difference is less than seven days, nothing will happen. If it is seven days or more, the update program will ask you if you would like to check the Web.

If you choose to do so, the program will check our website for the latest version. If it finds a version that is newer than yours, it will download a descriptive file for you to read. You can then choose whether or not to download the latest release.

The *read.me* file will also be downloaded, so that you can check back to see what new features have been added.

If you wish, you can tell Stat/Transfer to do an immediate check for a newer version rather than wait for an automatic check. To do so, go to the **About** tab and select **Right Now**. The update program will then check our website for updates as described above.

Uninstalling

Prior to uninstalling Stat/Transfer, you should [deactivate](#) it, so that you have an activation available if you want to use it on another computer.

Uninstalling Stat/Transfer

- On *Windows* select the **Uninstall** option from the **Start** menu's **Stat/Transfer** folder.

- On *OS-X* select the **Uninstall** item in the **Stattransfer16** folder of your **Applications** folder.

- On *Linux* or *Solaris*, go to the Stat/Transfer program directory and execute the **Uninstall** program.

- On *networks* Stat/Transfer can be uninstalled by executing **uninst.exe**, located in the **program** directory. If you would like the uninstall to be silent, without prompting, use the /S switch.

Technical Support

Before you seek support, please check the online help or look in the online manual and see if the solution to your problem can be found there. Be sure to check the "Frequently Asked Questions" section.

Also please check our help center at https://stattransfer.com/help

If you have a problem that you cannot resolve by these methods, the best way to seek help is by using our web forms at https://stattransfer.com/support/help. We have found that the use of these forms, rather than a free-form email, allows us to provide faster and better support. If you use a form, we are much more likely to get the information we need to address your problem on the first try. In addition, your request will be routed to the right person and automatically entered into a tracking system so that your problem will not fall through the cracks. To ensure the best service, please be sure to follow the instructions on the form and enter all of the information that is called for.

You can also seek support by using the **Log** tab. This method is particularly helpful if you think you have found a bug in Stat/Transfer, because it is possible to automatically send us a compressed and encrypted copy of the input file that was causing you problems, as well as a complete description of your environment and your own description of the problem. If you send a support request through this method, it is a very good idea to also use the form on our website. This will ensure that your problem is entered into our tracking system.

It is always good to make sure you are running the latest version of Stat/Transfer. You can go to the **About** tab and set **Check for Updates** to **Right Now**.

The Stat/Transfer User Interface

Starting Stat/Transfer

On Windows, the installation procedure will install a folder for Stat/Transfer in the **Programs** menu and a shortcut to Stat/Transfer. A shortcut to Stat/Transfer will also be installed on your desktop. Click on either of the shortcuts to start the program.

On OS-X, click on Stat/Transfer in the **StatTransfer16** folder of your **Applications** folder.

On Linux, a shortcut to Stat/Transfer will be installed on your desktop.

Transfer Dialog Box

Selecting the Input File Format

The input file format is selected in the first line of the **Transfer** dialog box, the **Input File Type** line.

Click on the **Input File Type** control arrow and you can browse through the list of supported file types. Select an input file type by clicking on it. The file type will be entered in the **Input File Type** line.

The ? Button
You can obtain information on a given file type by clicking on the '?' button.

The File Specification Line

Input File
The input data file is chosen using the second line of the **Transfer** dialog box, the **File Specification** line.

If your files are named using the standard file extensions for Stat/Transfer, given below, you will ordinarily use the **Browse** control to select a file.

When you click on **Browse**, a standard **File Open** dialog box will open. To select the input file, first make sure that the path is the correct ones for your input file. If not, change to the correct one.

Next, you need to select the correct file. Note that a wildcard file specification, '*.*ext'* has been created for the **File Name** entry, where '*ext'* is the Stat/Transfer standard extension for the type of input data file you have selected.

All of the files in the current directory with this extension will appear in a list box below the **File Name** line. Either use this list and click on the name of the file you wish to use or type the name on the **File Name** line.

The ? Button
If you click on the '?' button beside the **File Specification** line, you can obtain information on the file type currently displayed.

International Character Sets
If you have a problem reading international character sets, please see the **Encoding Options** entry in the **Options** dialog box.

Standard File Extensions

1-2-3	wk*
Access	mdb
ASCII - Delimited	txt, csv
ASCII - Delimited with Schema	stsd (Schema file)
ASCII - Fixed Format	sts (Schema file)
BayesiaLab	csv
dBASE and compatibles	dbf
DDI Schemas	xml
Epi Info	rec
EViews	wf1

EXCEL	xls
FoxPro	dbf
Gauss	dat
GenStat	gwb
HTML	htm*
JMP	jmp
JSON-Stat	json
LIMDEP	lpj
Matlab	mat
Mineset	schema, sch
Minitab	mtw
NLOGIT	lpj
ODBC	[none]
OpenDocument Spreadsheet	ods
OSIRIS	dict, dct
Paradox	db
Quattro Pro	wq?, wb?
R	rdata
RATS	rat
SAS for Windows and OS/2	sd2, sas7bdat
SAS for the Macintosh and Unix	ssd01, sas7bdat
SAS CPORT	stc
SAS Transport Files	xpt, tpt
S-PLUS	[none]
SPSS Data Files	sav
SPSS Portable Files	por
SPSS "Syntax" and Data Files	sps
Stata	dta
Stata Program and Data Files	do
Statistica	sta
SYSTAT	sys
Triple-S	xml

Selecting Worksheet Pages

Whenever you select a worksheet as input, Stat/Transfer will check to see if multiple pages are present.

If more than one page is found, Stat/Transfer will display a **Worksheet Page** selection line below the input **File Specification** line of the **Transfer** dialog box. If your worksheet pages are named, as they are in Excel, for example, these names will be used. Otherwise, dummy names, 'Sheet*n*', will be displayed, where *n* gives the number of the page.

The name of the first page of the worksheet will appear on the **Worksheet Page** line and, unless you select another one, will be the page used as the input data set by Stat/Transfer. If the data you wish to use are on a different page, click on the control arrow and select the appropriate page from the list that appears.

The option **Concatenate Worksheet Pages**, reached by clicking on the **Options** tab and then on **Worksheets**, allows you to combine worksheet pages into a single output file. If you check this option, when you name one page, then all of the pages will be read and concatenated into an output file of any type. This option is appropriate if your worksheet file contains many sheets that are identical in structure. These can be then be combined into a single output file.

Selecting Tables for Access and ODBC Input

Whenever you select either an Access file or an ODBC data source as input, Stat/Transfer will display a **Table** selection line below the input **File Specification** line of the **Transfer** dialog box.

The name of the first table will appear on the **Table** line and, unless you select another one, will be the table used as the input data set by Stat/Transfer. If the data you wish to use are in a different table, click on the control arrow and select the appropriate table from the list that appears.

Selecting Members of SAS CPORT and Transport Files

Whenever you select a SAS CPORT or Transport file as input, Stat/Transfer will display a **Member** selection line below the input **File Specification** line of the **Transfer** dialog box.

The name of the first member will appear on the **Member** line and, unless you select another one, will be the member of the SAS file used as the input data set by Stat/Transfer. If the data you wish to use are in a different member, click on the control arrow and select the appropriate member from the list that appears.

Most Recently Used File Lists

If you often use the same input file, you can select recently used files from the **File Specification** pulldown menu.

By default Stat/Transfer maintains a list of the last ten files that have been opened (ten for each filetype). You can select any one of these files by first clicking on the control arrow of the **File Specification** input field to display the list and then clicking on the file you wish to use.

If you would like to store longer lists, you can do so by setting a higher number in the **User Interface Options** of the **Options** tab**.**

Data Viewer

Data Viewer
You can preview your input data by pressing the **View** button in the **Transfer** dialog box. Your data will appear in a scrollable grid. By default, the viewer is on

The data can be sorted by any variable by clicking on the variable name. You can navigate to any row by entering the row number in the **Navigate To:** box and then pressing **Go**.

Columns can be moved by clicking and holding the column heading and then dragging the column to the new location.

To return to the **Transfer** screen, press **Close Viewer**.

To set viewer options, go to the **Options** tab and click on Data Viewer Options. Here are a few options you might want to know about:

Long String Viewer
The data viewer will display long strings, international character sets, variable characteristics and value labels. The long string viewer allows you to see strings that are too long to be viewed at the current column width.

By default, the viewer is on. To use it, simply left click on the cell you want to examine and a viewing window will display the string.   To disable it, uncheck the **Show Long String Viewer** option in **Data Viewer Options** in the **Options** dialog box.

By default, the viewing window will close automatically when your cursor leaves the cell that contains the string you are viewing. To turn off this behavior, uncheck the **Hide Automatically** option.

Variable Info Viewer
By default, if you click on the variable name at the top of the grid, a **Variable Info Viewer** box will open that will show you the variable name, its type and, if available, its label. To disable it, uncheck the **Show Variable info Viewer** option in **Data Viewer Options** in the **Options** dialog box.

Hide Automatically
By default, the viewing window will close automatically when your cursor leaves the cell that contains the variable you are viewing. To turn off this behavior, uncheck the **Hide Automatically** option.

Variable Selection Indicator

When the input file has been specified, Stat/Transfer by default selects all of the variables for transfer. A message will appear below the input **File Specification** line, telling you that all of the variables in the data set have been selected and giving the total number of variables.

If you wish to transfer all of the variables of the input data set, you need do nothing more to specify them. If you want to select only some of the variables in the input data set, click on the **Variables** tab at the top of the **Transfer** dialog box. Check the boxes to indicate which variables you want to transfer.

Once you have finished selecting variables, when you return to the **Transfer** dialog box, the number of variables you have chosen will appear.

For a complete discussion of the **Variables** Tab, see Variables Tab.

Selecting the Output File Format

The output file format is selected in the third line of the dialog box, **Output File Type**. It is always advisable to give the input file type first, before selecting the output file type.

Click on the **Output File Type** control to obtain the list of supported file types and to scroll through the list. Select a file type by clicking on it.

The list of output file formats will be the same as the list of input file formats, but with more choices of version, and with the following exceptions:

- HTML tables and Mplus files will appear on the output format list, since they can be written by Stat/Transfer, although they cannot be used as input.

- OSIRIS files will not appear, since they are only read by Stat/Transfer.

- SAS CPORT files will not appear since they are only read by Stat/Transfer.

- JSON-Stat files will not appear since they are only read by Stat/Transfer.

- Conversions from one xBASE file type to another are not supported since the file formats of dBASE and FoxPro are identical. Thus if a dBASE file is chosen as input, then FoxPro will not appear on the output format list and vice versa.

Stata Output

You can output a stata file as either Standard Stata, or Stata (SE). You can choose the version by selecting **Output Options(1)** from the **Options tab**.

SAS Output
SAS V6, SAS V7-8, and SAS V9 will appear in the list of output files types. You can specify the platform you wish for the output by selecting **Output Options(1)** from the **Options tab**.

Delimited ASCII Choices
If you wish to write delimited ASCII files, you will see two choices in the list of output file types:

ASCII/text- Delimited

ASCII/text- Delimited (S/T Schema)

These are described in the section ASCII Files - Delimited.

Fixed Format ASCII Choices
If you wish to write fixed format ASCII files, you will see several choices in the list of output file types:

ASCII/text- Fixed Format (S/T Schema)
    …
SAS Program + ASCII Data File
    …
SPSS Program + ASCII Data File
     …
Stata Program + ASCII Data File

These are described in the section ASCII Files - Fixed Format.

Selecting the Output File

The output file name is given on the fourth line of the **Transfer** dialog box. Since Stat/Transfer supplies a default specification for the output file using the input file specification, it is important that you always specify the input file name before the output file name.

Default File Specifications
Once the input file is chosen, Stat/Transfer will construct an output, or destination, file specification which has the same path and name as the input file but which has the standard extension appropriate for the output file type. This name will appear in the fourth line, the output **File Specification**, of the **Transfer** dialog box.

Changing the Name
If you do not wish to use the default name supplied by Stat/Transfer but instead would like the destination file to have a different name or extension, you can either use the **Browse** control to call up the **Save As** dialog box, or you can type the name directly into the **Transfer** dialog box.

Changing the Directory - the Most Recently Used List
If you wish use a different (drive and) directory, you can type in the directory directly. However, Stat/Transfer maintains a most recently used list of the directories to which you have transferred files. You can retrieve this list (which will show the output file name that appears in the output **File Specification** edit box) by clicking on the down arrow to the left of the output **File Specification** box.

Overwriting Output Files
By default, Stat/Transfer will check to see if the destination file already exists and will warn you that an existing file is about to be overwritten. You can suppress this warning in the **General Options** section of the **Options** dialog box.

See also:

Table Names for Access and ODBC Output
Naming Members of SAS Transport Files

Table Names for Access and ODBC Output

Whenever you select either an Access file or an ODBC data source as output, Stat/Transfer will display a **Table** selection line below the output **File Specification** line of the **Transfer** dialog box.

The default name of the output table will be taken from the input file name. If you wish to use another name, type it in the **Table** line.

Naming Members of SAS Transport Files

Whenever you select a SAS Transport file as output, Stat/Transfer will display a **Member** selection line below the output **File Specification** line of the **Transfer** dialog box.

The default name of the output member will be taken from the input file name. If you wish to use another name, type it in the **Member** line.

Resetting Stat/Transfer

The **Reset** control at the bottom of the **Transfer** dialog box is used when you wish to do more than one transfer during a Stat/Transfer session.

Once a transfer has been completed, click on the **Reset** control and the input and output file specifications will be removed, while the input and output file types remain.

Since Stat/Transfer will supply a default specification for the output file using the input file specification, it is important that you always specify the input file name before the output file name.

If you wish to change the input and output file types for a new data transfer, it is advisable to change the input file type first and then the output file type.

Running the Program

The Transfer Button

When you have specified the input and output file types and names (with information on member or page, when needed) and, if you wish to, you have also specified information on variables and case selection, click on the **Transfer** button and the data will be transferred.

Simple Transfers

If you wish to transfer everything in the input data set and you use the output target types assigned by Stat/Transfer, you need only specify the input and output file types and names in the **Transfer** dialog box. You can then click on the **Transfer** button and run the job. You do not need to enter anything in either the **Variables** or the **Observations** dialog box.

Stopping a Transfer

While the data are being transferred, the **Transfer** button is labeled **Stop**. If you click on it, your transfer job will be aborted. This is useful if you start a lengthy transfer and then realize that something is amiss.

When the transfer is complete, a message will appear at the bottom of the **Transfer** dialog box indicating that the transfer is finished and telling you how many cases were transferred.

Generating a Stat/Transfer Program

You can tell stat transfer to automatically generate a program when you run a transfer. To do this, click on the **Options** tab and then select **Stat/Transfer Program Generation**. By default, automatic program generation is turned off.

The new program keeps all of the selections you entered in the dialog boxes. Every option you selected in the **Options** dialog box is translated into its equivalent command processor setting.

This enables you to reproduce your transfer operation either from the **Run Program** dialog box or as a batch job from the operating system prompt.

It also allows you to precisely document how your transfer was performed. The program can be viewed or edited in the **Run Program** dialog box.

Automatic Logging

Stat/Transfer always writes a log that is displayed in the **Log** dialog box. You can view it and, if you would like, manually save it to disk.

However, you can tell Stat/Transfer to automatically write a log file to disk every time you do a data transfer. The log file will contain detailed information about what has occurred during your transfer.

To turn on this feature, check the option, **Automatically write a log file** in the **Transfer Logging/Statistics** section of the **Options** dialog box.

By default the log file will go to the same directory as your output file and will be appended to an existing log file. Options in the **Transfer Logging/Statistics** section allow you can change the name and whether or not an existing file is overwritten.

Variables Dialog Box

Variable Selection

Automatic Selection of All Variables in the Data Set
When the input file has been specified in the **Transfer** dialog box, by default Stat/Transfer selects all of the variables for transfer. A message will appear in the **Transfer** dialog box below the input **File Specification** line, telling you that all of the variables in the data set have been selected and giving you the total number of variables.

If you wish to transfer all of the variables of the input data set, you need do nothing more to specify them.

Manually Selecting Particular Variables
If you want to select only some of the variables in the input data set, click on the **Variables** tab at the top of the **Transfer** dialog box. The **Variables** dialog box will appear with a list of all of the variables in the input data set.

When you highlight a variable, the variable label will appear in the box at the upper right

By default, all of the variables are selected. You can select or unselect variables one by one by going to a particular variable and toggling selection either on or off for that variable. To do so, click on the check box next to the name or click on the variable name and press the SPACE key.

If you wish to select or unselect a group of variables, use the **Quick Variable Selector**.

Quick Variable Selector
The box in the upper right corner enables you to specify selection criteria for the variables displayed in the list box at the left of the page. This is considerably less tedious for long lists of variables than manually checking or unchecking them.

To select or unselect all of the variables, type a star, '*', in the **Quick Variable Selector** box and click either **Keep** or **Drop.**

Selection conditions can take the form of the wildcard characters '*' or '?' or you can use variable ranges. The question mark matches exactly one character, while the asterisk matches more than one. Unlike standard wildcards, more than one asterisk can be included in a specification. For instance: '*inc*' will match any variable with the string 'inc' in any position. Ranges of contiguous variables can be specified with a dash

(without spaces) between two variable names. For instance 'distance-a9' will select (or drop) variables 'distance' through 'a9', inclusive.

Space or comma delimited lists of conditions can be entered at one time. For example:

```
factor1,cluster,a2-a10,L1*
```

followed by a click on the **Drop** button, will uncheck the variables 'factor1', 'cluster', 'a2' through 'a10', and any variable which starts with the string 'l1'.

If needed, you can successively refine your selection by entering conditions and then clicking on either the **Drop** or **Keep** buttons, or, alternatively, by manually checking or unchecking variables in the list box.

Variable Selection Indicator
Select all of the variables you want to transfer. When you have finished, you can click on the **Transfer** tab at the top of the dialog box and you will return to the **Transfer** dialog box, where you will see a message telling you how many variables have been selected.

Value Labels Browser

If your input file has value labels, the option **Value Label Browser** allows you to display them for each variable.

This option is set by clicking on the **Options** tab and then clicking on **User Interface Options**. You can choose to have the value labels displayed in a vertical box, just to the right of the variable names, or in a horizontal box below the variable names. The default is to display the value labels in a horizontal box.

Target Output Variable Types

Output Variable Types

Systems differ widely in the number and variety of variable types they support. When data are transferred from one file type to another, a variable type in the output format must be assigned to each of the variables being transferred.

Note that with Stat/Transfer, numerical precision is never lost in the transfer process, since all numerical variables are stored internally as double precision floating point numbers and are then written out according to the assigned variable type.

Stat/Transfer automatically reads your dataset an extra time in order to determine the optimum output types for each variable and the maximum length of string variables. In almost all cases it is appropriate to accept the output types that Stat/Transfer chooses. However, there are times that you may wish to override these defaults and set the output types manually.

Target Types Assigned by Stat/Transfer

When assigning default output variable types, Stat/Transfer attempts to use all of the information at its disposal about the input data variables in order to preserve numeric precision and, at the same time, minimize the size of the output data set.

When reading numerical variables, Stat/Transfer selects a target output variable type based on the information available to it. This target variable type is not used for internal storage during the transfer, but is simply the preferred output type. If this type is not supported in the chosen output file type, the best approximation will be chosen.

The various target output variable types used by Stat/Transfer are:

byte	One byte signed integer (-128 to 100)
int	Two byte signed integer (-32768 to 32740)
long	Four byte signed integer
float	Four byte IEEE single precision floating point number
double	Eight byte IEEE double precision floating point number
date	Date stored as serial day number (days since December 30, 1899)
time	Fraction of a day (12:00 noon = .5)
date/time	Floating point number (integer - serial day number; fractional part - time)
string	Character string (stored internally in Unicode UTF-8 representation.

Remember that the target type will not necessarily be the actual output type. If the target type assigned to a variable by Stat/Transfer is available as one of the variable types

of the output file format, then that type will be used for the output. If the assigned target type is not one of the available output types, then a format of the next larger size will be used.

Automatic Optimization of Target Types

Stat/Transfer attempts to produce the smallest possible output data set. After reading your data, Stat/Transfer will make an optimization pass during the transfer, to determine more information about each variable and then will assign target output types.

First, before the transfer, information from the data file dictionary is read. Unfortunately, for some input data types, this information is not sufficient to do anything other than set all of the output variable types to 'float'.

Stat/Transfer will then make an additional optimization pass through your data. By default this will occur during the transfer. This pass will only be performed if the file contains any string variables or if the selected output file type is such that the output file could be made smaller by optimization. Some output file types, such as Stata, have a rich assortment of storage types and benefit from optimization. Others, such as worksheets or files that have only one numeric type (SPSS for example), do not benefit. Optimization must be performed for any file with string variables, because the length of output strings can potentially change as they are transcoded to different output encodings.

Stat/Transfer can determine whether any variables can be represented as integers, and, for those cases, it can determine the smallest possible integral type that can be used to represent the data. Further, if a variable cannot be represented by an integral type, Stat/Transfer can automatically determine whether it can be represented by a float instead of a double without a loss of information. Information on the maximum length of string variables after conversion to the output encoding is also accumulated, so that truncation will not occur and to enable variables to be be stored in variables of the smallest possible length.

In almost all cases it is appropriate to accept the output types that Stat/Transfer chooses. However, there are times that you may wish to override these defaults and set the output types yourself.

Optimization, by default, takes place during a transfer. The **Optimize** button is used to optimize manually before the transfer. This allows you to see the target types that Stat/Transfer has chosen for your variables before they are transferred, and to change any output types that you wish. This is discussed in the section Changing the Types of Output Variables.

Manually Changing the Types of Output Variables

In most cases it is appropriate let Stat/Transfer automatically optimize your variables and to accept the output type that Stat/Transfer chooses. However, there may be times when you wish to specify the output types for some variables.

Changing Target Output Types
The **Variables** dialog box displays a list of all of the variables in the input data set. When the input file is opened, Stat/Transfer assigns target output types based on information in the data file dictionary and displays them next to the list of variables. These will not be the final target output type assigned by Stat/Transfer. The final output type will be assigned automatically during the optimization pass.

You can change the output variable types by using the **Optimize** button on the right side of the **Variables** dialog box after you have selected all of the variables that you wish to transfer.

When you press the **Optimize** button, the optimization step will take place before the data are transferred. After the optimization step, you can examine the target types that Stat/Transfer has chosen for your variables and change any that are not appropriate.

After you press the **Optimize** button, to see the final target output type, choose one of the variables in the list box on the left and the output type automatically assigned by Stat/Transfer will be displayed on the buttons on the right.

If you wish to change the output type for a particular variable, simply click on the new type you want to assign to that variable.

Quick Variable Type Changer
This feature is useful when you want to change the type of a number of variables, where the new type is the same for all. The **Quick Type Changer** dialog box enables you to specify selection criteria for the variables. This is considerably less tedious for long lists of variables than manually changing each one.

Selection conditions are entered in the same way as in the **Quick Variable Selector** box. They can take the form of the wildcard characters '*' or '?' or you can use variable ranges. The question mark matches exactly one character, while the asterisk matches more than one. Unlike standard wildcards, more than one asterisk can be included in a specification. For instance: '*inc*' will match any variable with the string 'inc' in any position. Ranges of contiguous variables can be specified with a dash (without spaces)

between two variable names. For instance 'distance-a9' will select (or drop) variables 'distance' through 'a9', inclusive.

Space or comma delimited lists of conditions can be entered at one time. For example:

```
factor1,cluster,a2-a10,L1*
```

will select the variables 'factor1', 'cluster', 'a2' through 'a10', and any variable which starts with the string 'l1'. The output type for these variables is specified from the drop down menu below the line specifying the variables.

If needed, you can successively refine your selection by entering conditions and then clicking on either the **Drop** or **Keep** buttons, or, alternatively, by manually checking or unchecking variables in the list box.

Automatic optimization will, of course, not take place during your transfer when you have used the **Optimize** button.

Limitations on Changing Output Variable Types
Output variable types can be changed freely for ASCII files and worksheet files. For all other file formats, you can change freely among the numeric types of 'byte', 'integer', 'long', 'float' and 'double' and you can change among the time types. However, conversions between any of the numeric types and dates or strings are not supported.

You should be careful not to choose a smaller type than that chosen by Stat/Transfer unless you are sure you know more about your data than Stat/Transfer does.

Remember that you are selecting a "target" type. If the output data format does not support the specific type you have selected, then Stat/Transfer will use the best match to the type you have selected.

You can determine the output variable types supported for each output file type by consulting the table given in the topic describing that file format.

Handling Mixed Data
If you have mixed data in which some variables need doubles and others do not (for example, you might have precisely measured dollar amounts, which should be in doubles, along with scales of survey items, which should be in floats) you should press the **Optimize** button in order to designate integers for the right variables and then designate floats and doubles to reflect the appropriate level of measurement for each variable.

Automatic Dropping of Constants

You can tell Stat/Transfer to automatically drop variables that are constant or missing for a selected subset of data. You select this option by checking the **Drop Constants** check box in the **Variables** dialog box and then pressing the **Optimize** button.

This feature is useful when the part of a data set selected for transfer contains variables with values that are either constant or missing (such as a pregnancy variable when only male subjects are selected or variables in yearly surveys where the same questions do not appear for each year.)

This feature is not likely to be used often, but is extremely valuable when it is needed, since if the data set has a large number of variables, it can be exceedingly tedious to select only the meaningful ones manually.

Use Doubles Option

The **Use Doubles** option tells Stat/Transfer whether or not to put variables with fractional parts into 'double' or 'float' on output. The default is now to have Stat/Transfer use doubles.

If you do not change the default behavior, Stat/Transfer will evaluate each variable to see if it can be represented as a 'float' without a loss of information and will put only those variables that require it into a 'double'. This is the safest option and is now the default.

However, most data are not measured with more than eight or nine digits of precision (survey data, for example, never are). Therefore, if you need to save space on output and do not need more precision, you can change the default behavior, so that only 'floats' are used.

You can reset the **Use Doubles** option by clicking on the **Options** tab, selecting **General Options** and uncheck the **Use Double** box.

Alternatively, for a single transfer, you can uncheck the **Use Double** box at the bottom right of the **Variables** dialog box and then press the **Optimize** button.

Observations Dialog Box

Selecting Cases

To reach the dialog box that allows you to select specific cases or records from your data set, click on the **Observations** tab at the top of the dialog boxes. This will bring the **Observations** dialog box to the front.

The scrolling text box in the upper left corner provides brief, on-screen documentation on how to select particular data records based on conditions that you specify. The variables of the input data set are listed in the box at the right of the screen.

At the bottom of the screen is the case-selection field in which you enter the case selection, or WHERE, expression that will specify cases. This expression gives the conditions on the variables that will define the subgroup of the data set that you wish to select.

Variable names can be entered in this field by selecting their names from the variable list box. When you double-click on a variable name it will be copied to the case-selection box.

Case-Selection Expressions
The WHERE statement is used to give the conditions on the variables that will define the subgroup of the data set that you wish to select.

The case-selection, or WHERE, expression, has the following form:

WHERE   *variable expression*   *relational operator*   *selection condition*

Here, *variable expression* consists of a single variable or an expression involving several variables, *relational operator* is one of the operators listed below, while *selection condition* gives specifications for the variables to be selected.

Variable Expression
All of the usual arithmetic operators [+ - / * ( )] are available for use in this expression.

If variable names used in WHERE expressions contain embedded blanks or characters such as relational operators or arithmetic like '/', then they must be enclosed in single quotes.

<u>Internal Variable</u>

An internal variable, '_rownum' is available which allows specific rows or records of the data set to be referenced.

Relational Operators
The following relational operators are available:

=	equals
!=	not equal
<	less than
>	greater than
<=	less than or equal
>=	greater than or equal
&	and
|	or
,	or (used in a series)
|	not

The modulus operator is also available:

%	remainder after division by the operand following

Selection Conditions
If variable values consist of string, then when they contain blanks or characters such as '/', they must be enclosed in double quotes.

Examples
Examples of selection conditions given by WHERE expressions are:

```
where educ = 12 & rate > .2

where (income1 + income2)/famsize < 20000

where income1 >= 20000 | income2 >= 20000

where acct != 2001

where name = smith

where 'dept-sales' = "auto loan"
```

```
   where  id  %  2 = 0    (which selects all even values of
'ID')

  where _rownum < 200 (which selects rows 1 - 199)
```

Using Dates

Dates, times, and date/time values can be used in expressions by enclosing the operand in square brackets. Stat/Transfer will use the masks in you have set in your Date Formats/Reading options to scan and convert the values. For instance

```
  where admission_date > [01/05/2015]
  where time > [1:00 pm]
```

Wildcards in Selection Conditions

Wildcards (* or ?) are available to select subgroups of string variables. For example:

```
   where account = ?3*

  where name = mc* | name = mac*
```

Note that when the wildcard '?' is used, it replaces a single character, while the wildcard '*' replaces an unspecified number of characters. Thus the specification '?3*' will select account numbers of any length that have a three in the second place.

Comma Operator

The comma operator ',' is used to list different values of the same variable name that will be used as selection criteria. It allows you to bypass potentially lengthy OR expressions when selecting lists of values. For example, the WHERE expression above can be more easily written:

```
   where name = mc*,mac*
```

Other examples are:

```
   where age = 21,31,41,51,61
```

which will select only the listed ages, and

```
   where caseid != 22*,30??,4?00
```

which will select all cases except those id's starting with '22', or four character id's starting with '30', or starting with '4' and ending with '00'.

Preserving WHERE expressions

Ordinarily the WHERE expression is cleared after a transfer operation. If you wish to apply the same expression to several input files, you can check the box **Preserve expression between transfers** and your expression will be available for re-use or editing for your next transfer run.

See also:

Missing Values
Sampling Functions

Missing Values

You can test to see if the value of any variable is missing by comparing it to the special internal variable '_missing.'

For example

```
where income != _missing & age != _missing
```

Sampling Functions

Three functions are available for sampling.

Random Samples
The first function

   samp_rand(prop)

allows for simple random sampling. Each case is selected with a probability equal to *prop*.

For example, for a random sample of one tenth of a data set, use:

```
where samp_rand(.1)
```

Random Samples of Fixed Size
The second function

```
samp_fixed(sample_size,total_observations)
```

allows a random sample of fixed size to be drawn. When using this function, the first case is drawn with a probability of *sample_size/total_observations*, and the succeeding *i'th* case is drawn with a probability of *(sample_size - hits) / (total_observations - i)*.

For example, if you had a data set of 1000 cases and wished for a random sample of 25 cases, you would specify:

```
where samp_fixed(25,1000)
```

Systematic Random Samples
Finally, a third function

   samp_syst(interval)

performs a systematic sample of every *n'th* case after a random start. For instance, to take every 6'th case, use:

```
where samp_syst(6)
```

Sampling Subsets of the Input Data
Expressions are evaluated from left to right. You can thus sample from a subset of your cases by subsetting them first and then sampling.

For example, to take a random half of high school graduates, use:

```
where schooling >= 12 & samp_rand(.5)
```

Sampling Seed and Reproducible Samples
The random number generator that provides the basis of these sampling routines is
'rand_port()' in Jerry Dwyer, "Quick and Portable Random Number Generators." *C
Users Journal*, June, 1995, pp. 33-44. By default, it is seeded using a permutation of
the time of day, and will yield a different sample on each run.

If you need a reproducible sample, you can generate it by using the same seed each time.
The seed is entered in the **General Options** section of the **Options** dialog box and
should be a positive integer in the range of one through 2,147,483,646.

Options Dialog Box

General Options

Ask Permission before Overwriting Files
The option **Ask Permission Before Overwriting Files** is on by default. If a file, or a database table exists, you will be prompted for permission before it is overwritten.

If you wish to suppress these warning messages, click on the box to remove the check mark.

Write New, Numeric Variable Names (Vn)
When you go from one format to another, by default Stat/Transfer will create legal variable names for you, based as much as possible on the original names. In particular, when you transfer from systems such as Paradox or JMP, which allow long variable names with embedded spaces, to older systems which restricts variable names to eight characters, by default Stat/Transfer will truncate for you. However, these truncated names often have little resemblance to the names you started with. Stat/Transfer will use the variable names as variable labels, so that your original names are available.

If you check the option **Write new, numeric variable names. (Vn)**, instead of the default variable names, Stat/Transfer will create new variable names of the form V1...VN. This is chiefly useful when dealing with truncated names. If your output system supports variable labels, it is sometimes better to check this option and have Stat/Transfer simply create numeric names for your variables. You can then use the variable labels for the description.

Because this option is likely to be useful only in special circumstances, it reverts to the default between sessions.

Preserve value label tags and sets
Many software packages allow users to assign the same set of value labels to more than one variable. (In SAS, the term for value labels is "user-defined format"). For example, a survey with a list of questions with "Yes" and "No" responses could use the same set of value labels for the variables associated with each of these questions.

If the option **Preserve value label tags and sets** is checked, the mapping of value label sets to multiple variables will be preserved on output. If tags are used in the input file to identify value labels sets, these will be preserved. Otherwise, tags will be constructed by Stat/Transfer (LABA-LABZ and so on).

If this option is not checked, each labeled variable will have a unique value label set and the tag used to identify the set will be constructed from the name of the variable.

This option is off by default.

Seed for Sampling Functions

By default, the sampling functions in WHERE expressions will generate a starting seed randomly, based on the clock time. This means that each time you run a transfer on a given file you will select a different sample. If, in contrast, you need a reproducible sample, you can enter a seed for the random sampling process. The seed should be a positive integer in the range of one through 2,147,483,646.

Variable name case conversions

Stat/Transfer always follows the variable-naming rules of the output file type and will convert input names so that they will conform to those rules. Some older packages require upper case variable names. Other more modern packages allow mixed case variable names. Some packages, notably Stata, S-PLUS and R allow mixed case variable names, but are case sensitive. In these case-sensitive systems, if Stat/Transfer were to move data from an upper-case system and do no case-conversion, the user of the data set would need to always hold down the shift key when typing in variable names.

Stat/Transfer allows you to specify your case conversion preferences for case-insensitive packages as well as case-sensitive packages.

The available options are given in the drop-down menus for both case-sensitive and case-insensitive programs and are:

- Convert to lower
- Preserve if mixed case
- Preserve always
- Convert to upper

The default for case-insensitive programs is "Preserve always".

The default for case-sensitive programs is "Preserve if mixed case". This choice is designed to handle mixed case variable names such as "FamilyIncome". If both upper and lower case letter are found in a variable name, it will be left untouched. Otherwise it will be converted to lower case.

Formats which are considered case-sensitive on output are: Stata, S-PLUS, R, and Matlab.

Formats that require upper-case variable names are: SAS version six, SAS XPORT, SPSS Portable, SYSTAT, Epi Info, and older xBASE versions such as Clipper and dBASE II.

All other formats are case-insensitive.

Optimization

By default, Stat/Transfer will optimize data whenever it is necessary. This will assure that text variables will not be truncated and that the storage of numeric output data will be as compact as possible.

For some output formats, such as fixed format text, this optimization step is absolutely necessary. *We strongly recommend that you just let Stat/Transfer do its work and not attempt to override or modify the optimization process.*

You should only modify the options which control optimization if:

1. you know what you are doing AND...
2. your input dataset is so large that the time you may save is worth the risk of corrupted or less-than-optimal data.

That said, there are options that allow you to control the optimization process.

How Stat/Transfer Optimizes

The Stat/Transfer optimization process makes an additional pass through the data before transferring it. For each variable, in each selected case, running statistics are accumulated. The following calculations are performed:

Single or Double Width?
For numeric variables, the minimum and maximum values are recorded. Each value is then tested to see whether it has a fractional part. If so, it is tested to see if it can be represented in a single precision floating point number or if it must remain in double.

Fixed Width
For output formats that are essentially fixed-width, formatted text, it is necessary to determine the maximum width of each numeric variable. To do this, they are written to memory to determine their width. This is only done for output formats that require it, but for those formats it is mandatory.

Transcoding Strings
String variables are transcoded into the output encoding and compared to their stored maximum encoded length, which is adjusted as necessary. Converting from one character set to another can change the string width. For example: if a single-byte character set (such as Greek or Russian) is transcoded to UTF-8, the output string width could be

twice as long as the input. Thus, when there are any strings in the input file, optimization is usually needed.

Final Step

After all cases have been read, the target types for integers are determined based on their maximum and minimum values. For any variable that is not integral, the target type will be either double or single precision floating point. The width of strings in the output file will be determined by their maximum width, as will the width of numbers which are going into fixed format ASCII file types.

Optimization must be performed:

- When the input file format gives no information of the widths of variables (e.g. CSV, Excel) and the output format needs the width of output variables (This includes almost everything but .csv files and worksheets).
- When the output file requires the calculation of the formatted width of numeric variables (fixed format ASCII, dBASE).

If optimization is required, you will be able to set options to suppress it, but they will not be honored. Messages about optimization can be found in the log if the log-level is set to "information and errors" or higher.

If Stat/Transfer detects any variable truncation when optimization has been turned off or limited, the transfer will be halted with an error message.

Guide to Optimization Options

Number of records to optimize
The possible values for this are:

All (the default) – Optimization will be performed on all cases whether it is strictly necessary or not. This is the recommended setting and for datasets of reasonable size, this will take very little time and will result in an output file that is guaranteed to be correct and as small as possible

None - This is not recommended and should only be used if your dataset is extremely large and you are confident that optimization is not necessary.

Number – This setting is a compromise and is sometimes sensible if your dataset is very large and you are confident that the first n cases will fairly represent your data.

Preserving Widths
If the Number of Records to Optimize is set to "all" or to a specific number, you can exert some control over the widths of both string and numeric variables.

Preserve String Widths if Possible

Normally, when optimizing, Stat Transfer will calculate the minimum string width for each variable. This ensures that the output file will be as small as possible. This option allows you to maintain the input string width. This is particularly useful when combining different files. If this option is checked, Stat/Transfer will use the input width will be used as the output width if it can do it without losing data. More precisely, the variable width will be the minimum of the transcoded string width and the input width. The output can be greater, but not less than the input width. For plain ASCII data, it will be the same.

Preserve Numeric Widths if Possible

This option is useful if you are reading text data or data which comes from a format which preserves the width originally used to read the data (SPSS is a prime example). If you are writing text data or want a variable to be formatted with the same width as was present in the input, check this option. If, after optimization, the variable is wider than the input width, the variable will be widened to prevent loss of data.

Special Option for Stat/Transfer Schemas

If you want total control over your variables and you know what you are doing, you can put the keyword

NO-OPT

In your Stat/Transfer schema. If it is present, your data will not be optimized no matter what output format is chosen.

User Missing Values

User Missing Values
You have some control over the way missing values are treated for input files containing more than one type. At present the **User Missing Values** options apply to SPSS files (both Data and Portable), Stat/Transfer Schemas (if the user missing options are used) and OSIRIS. The options are selected with the buttons **Use All**, **Use First** or **Use None**.

Some statistical systems distinguish between "system missing," such as the result of a divide by zero, and "user-missing," a numeric value which is defined as a missing value by the user. Further, particularly in survey research, distinctions are made between user-defined missing values that represent structurally missing data (such as answers to pregnancy history questions from male respondents), and those that represent categories of non-response or simply the failure of the interviewer to properly collect the data.

Conventionally, zero is used to represent "inapplicable" missing values, and higher numbers are used to represent such responses as "don't know," "refused" and "not ascertained". While inapplicable data is analytically equivalent to "system missing", there can be legitimate research interest in the patterns of non-response represented by the other categories of missing data.

Use All By default, when multiple missing values are allowed (as in SPSS, for example) they are mapped into a single missing value on output. This corresponds to selection of the **Use All** button.

The mapping to a missing value on output is determined by the option **Map to extended (a-z) missing**.

Use First If you select **Use First**, the first user-defined missing value will be mapped to the system missing value and the rest will be transferred intact to the target data set. **Use First** will often be the most useful of the options, since it will allow tabulations in the target package of patterns of non-response.

Use None If you choose **Use None**, then all of the user-defined missing values will be transferred retaining the input value in the target data set.

Map to extended (a-z) missing
By default (when this option is left unchecked), all user missing values that are selected according the options for user missing values (**Use All/ Use First/Use none**) will go to a single missing value which will then be converted to the "system" missing value in the target package ('.' in SAS or Stata, for example).

If the option **Map to extended (a-z) missing** is checked, user missing values will be mapped, if possible, to extended missing values in target packages that support them (SAS, ASCII, or Stata).

By default, this mapping will be done in order, with the first missing value going into .a, the second into .b, and so on.

Map using variable labels
If the option **Map using variable labels** is checked, the first letter of the value label will be used as the missing value. For instance, if the value '0' is a user missing value and is labeled as "inapplicable", it will be mapped to '.i'. This mapping will only occur for missing values that are computed with an equal operator.

If there is no label, or if the missing letter has already been used, the missing value will then be mapped sequentially to '.a' - '.c'.

Since current versions of Stata and SAS support the labeling of missing values, this option is less useful. If you have value labels, it is best to keep it unchecked and rely on the value labels to differentiate your values.

Note that we believe these options are potentially dangerous. To avoid the chance of users checking one of these options and then forgetting about it, Stat/Transfer does not save the settings when options are automatically saved at the end of a session.

Date/Time Formats – Reading

Stat/Transfer version 16 introduces a new and much easier way of specifying formats or masks for reading dates and times. These are similar to, but easier to use than those in Stata. They are easier to use because the formats are implemented so they simply skip over non-date components and punctuation. For instance, the date mask YMD can read all of the following dates:

Jan 1, 2022
January 01, 2022
1/1/2022
02/02/22
1-1-2022
January 20th 2022

Mask Element	Comments
Y	Year – this will read two or four digit years. If two digits are found the "century crossover year (by default 30) be used to set the century component
M	Month – this will read one or two digit month numbers and also spelled-out month names and abbreviations. The names and abbreviations, by default, are in English, but if you set the ASCII/Read option "Use localized Month Names", Stat/Transfer will use the names that are set in your computer's locale
D	Day – the day of the month.
h	Hour – 0 to 23 or 1 to 12 if AM/PM (or its local equivalent) is present.
m	Minutes - 0 to 59
s	Seconds – 0 to 59. Fractional seconds (tenths, hundreds and milliseconds) will be read if a decimal point is present.

Notes
Spaces are not significant – Y M D is equivalent to YMD. The elements are read from left to right and compared to the input. Spaces and punctuation, as well as extraneous characters are ignored in the input as well.

Examples

Input Date	Mask
January 1, 2022	MDY
11:01	hm or hms
11:01.22	hms

22:01.001	hms
11:30 PM	hm
01012022	MDY
01-02-2022	MDY
01-FEB-2022	DMY
2022-01-11T12:43	YMD hm
06Jan2021	DMY
6Jan2021	DMY
20210111	YMD

Century Changeover Year

When you are reading two-digit years, some may fall in the twenty-first century and some in the twentieth. You can use this option to control how two digit years are read. Any two-digit year less than the changeover year will have the first two digits of the complete four digit year set to 20. Any year greater than or equal to the changeover year will have the first two digits set to 19.

The default for the option is '30', so that the changeover year from one century to another is 1930. Thus the date 1/1/21 will be interpreted as January 1, 2021, while the date 1/132 will be interpreted as January 1, 1932.

If your data refer to dates earlier than 1930, such as birth dates, you will need to override the default behavior and specify a different changeover year. If, for example, you specify '00', this would cause all two digit dates to be interpreted as years in the twentieth century.

Date/Time Formats - Writing

Stat/Transfer gives you considerable control over how dates and times written to output ASCII files. You can control the formatting for date values, time values and combined date/time values in the **Date**, **Time** and **Date/Time** edit boxes.

Output formats that you supply are used to convert date and time values to character strings. Each date or time part of the output format has the form '%char'. Leading zeros cause the value to printed with leading zeros. For example '%0d' will print the day of the month with a leading zero.

The characters below are used to create the output formats. Anything to be printed in the output character string that is not in the list below, such as commas, spaces or other delimiters, must be given explicitly in the output format.

%a	abbreviated weekday
%A	full name of weekday
%b	abbreviated name of month
%B	full name of month
%d	day of the month (1 - 31)
%D	day of the year (1 - 366)
%H	hour (24 hour clock) (0 - 23)
%I	hour (12 hour clock) (1 - 12)
%m	month as number (1 - 12)
%M	minutes (0 - 59)
%N	milliseconds (0 - 999)
%1N	tenths of seconds (0 - 10)
%2N	hundredths of seconds (0 - 99)
%p	'am' or 'pm'
%S	seconds (0 - 59)
%y	year as two digits
%Y	year as four digits

%%	% character

The default formats for converting dates and times to strings are:

Date: %m/%d/%Y   (5/18/1945)
Time: %0H:%0M:%0S   (14:05:48)
Date/Time: %m/%d/%Y %0H:%0M:%0S   (10/1/1990 02:20:09)

Encoding Options

Introduction to Character Encoding

In the early days of computing, characters and numbers were represented by seven (and sometimes fewer) bits. The most popular of these encodings, ASCII, allowed the representation of upper and lower case letters, numbers and various symbols. Later, different eight-bit relatives of ASCII were developed, which allowed the encoding of an additional 128 characters, such as accented letters and various other symbols. For languages such as Japanese, which required many more characters, other multi-byte character sets were developed. This resulted in different, incompatible character encoding systems, even for the encoding of a single language's character set. Fortunately, order is now being restored to this Babel with the proliferation of Unicode, a means of uniformly representing the characters of all languages.

In Stat/Transfer 13, all characters are represented internally in Unicode. It is thus capable of handling strings in any language. However, many of the file formats that Stat/Transfer supports were written in the days in which ASCII was *the* way of representing strings. Thus, when read, they will not give Stat/Transfer information on the code-page that is in use and when they are written, they have no place for Stat/Transfer to store the code page that was used.

For some file formats, the user will not need to worry about analysis of this. Excel '97 and above, for instance, stores characters in Unicode. If you are going to Access or SAS Version 9 or above, no conversion will be necessary as both of these are Unicode aware.

Your computer's operating system maintains a setting of the current working code page. In most cases applications will use that information to encode characters in a consistent manner. However, if someone in Japan, or Greece, or Russia sends you a file, you will need to tell Stat/Transfer the encoding that was used to write their files.

The options on this page provide a way for the user to specify, when necessary, the encoding that is present in an input file and the one that is desired in an output file.

Input Character Set

In general, unless you are sure of what you are doing, you should leave this option at the default setting, **Use current system default code page.** If that is checked, you will be able to see which code page is in use on your computer. That code page will then be used for files that that do not have known encoding information. If the encoding of an input file can be determined from the contents of the file (or its format), that will, of course, override this setting.

If you choose to override the default behavior, you can choose a different encoding by first selecting the **Region** and then the **Character set.** Because Stat/Transfer represents characters internally in Unicode, any character set can, in principle be converted on input. However, if you select an incorrect character set, what you will get is likely to be nonsense. Therefore we strongly urge you to look carefully at your data in the viewer to make sure that all is well and to make sure that the font you have selected for the viewer is capable of displaying the characters you need.

Output Character Set

Unless you are sure of what you are doing, you should leave this option at the default setting, **Use current system default code page.** This is especially true for the output character set because any input character set can be converted to our internal representation, Unicode. However, you have to get it just right when going from Unicode to another single or multi-byte character set or you will be guaranteed to get total nonsense. For instance, if you have read a Japanese file and then want to convert it to a multi-byte Chinese character set, Stat/Transfer will simply stop in its tracks since this is impossible (unless the only characters consist only of numbers, simple punctuation, and the letters A-Z.)

For file formats that are Unicode-aware (e.g. Excel and SAS 9+), Stat/Transfer will write Unicode, regardless of how you set this option.

On Encoding Errors

Unicode can represent any character. Unfortunately the same cannot be said for other character sets. When Stat/Transfer moves character data from its internal Unicode representation to a character set that can represent fewer characters, there is some probability that some will not fit. For example, Microsoft applications such as Excel store characters in the Unicode representation and in some cases, although it looks like the contents of the data can be represented in a Western European character set, there are some characters that cannot be encoded. These include the right and left apostrophes and, often, the Euro sign.

The default behavior when this occurs is Substitute. With this option, Stat/Transfer will substitute characters for those that cannot be converted. For example, it will substitute a single quote for right and left apostrophes, and, if necessary, non-accented letters for those with accents. If no substitution is possible, underscores will be substituted. If you do not want any substitution performed, you can check the option Stop, which directs Stat/Transfer to stop on the first conversion error encountered.

The error limit will limit the number of permitted substitutions. The default is 100.

Note: If your Western European data have Euro and other currency signs, a good choice for your single byte output character set is ISO-8859-15, which is a more modern version of ISO-8859-1.

ODBC/Access Options

The following options allow you to fine-tune your ODBC transfers. They are generally for advanced users.

Use NULL instead of empty strings
A null string is a string for which a value has never been entered while an empty string has zero length. Most databases support both NULL and empty strings (Oracle is the exception which converts empty strings to NULL strings). If you check this box, Stat/Transfer will write NULLS instead of empty strings into your database.

Prefer "datetime" over "smalldatetime" for MS SQL/MySQL
The 'smalldatetime' type in MS SQL Server represents the dates between Jan 1,1900 to June 6, 2079 with accuracy to a minute. If this is sufficient for your needs and you want to save space, uncheck this option. The 'DateTime' type, in contrast, represents time values to an accuracy of close to 3 milliseconds and dates back to 1753.

Prefer (W)VARCHARover (W)CHAR
On output, if your data are stored in 'char' or its Unicode variant 'wchar', the length of the field will be equal to the length of its widest member. On the other hand, 'varchar' is stored in variable length fields that save storage when the length of the string data varies between cases. In general, you will want to leave this option at its default and write variable length string types when these are permitted.

Show the name of the table/view owner
A table or view in a database is identified by its owner and its name. If you are in an environment in which tables with different owners have the same name, you will need to check this option to make sure that you can select the proper table.

Rows to read when scanning datetime variables
In some databases (including Access) there is only one date type, which can hold dates, times, and date/time values. Stat/Transfer will, by default, read ten rows of data to determine the type of such variables. If ten rows are not enough for your data, you can set a higher number here.

Append to Access and ODBC tables
This option (which is off by default), allows you to append your data to an existing database table.

Stat/Transfer will match as many variables as is possible to those already in the table and add your data to the matching columns. Obviously at least one column must match

exactly and, in addition, the table must be free of constraints that would prohibit a simple 'append' operation, such as those requiring unique keys.

Drill Schema Name

When reading Apache Drill, if the name of the schema is not specified in the data source options, it can be specified here.

ASCII/Text Files - Read Options

Delimiter
This option will give you a list of possible delimiters for input ASCII files. You can choose to have Stat/Transfer automatically sense which delimiter to use or you can choose from the list: commas, tabs, spaces or semicolons. If you have a delimiter that is not on the list, click on 'Other' and enter the delimiter you wish to use.

Combine adjacent blanks
This option is available for space delimited files only. It is useful if data values are delimited by one or more blanks or tabs.

The default for **Combine adjacent blanks** is 'off'.   If you turn this option on, you can select 'Spaces' in which case multiple blanks are treated as one blank, or you can select 'Spaces and tabs', in which case multiple instances of tabs and blanks are converted to a single space.

Variable Names
By default, Stat/Transfer will sense whether the first line of your input data set contains field names or data. You may, if you wish, explicitly override this default.

AutoSense: If this option is set to 'AutoSense', Stat/Transfer will look at the first and second rows of data. If there is a change from a string to a number for one or more variables between these rows, Stat/Transfer will use the first row as the field names. This will fail if your first row contains the field names, and all of your variables are of the string type. In that case you should choose one of the following two options:

First Row: When this option is set to 'First Row', Stat/Transfer uses the data found in your first row as the field names.

Make Up: When this option is set to 'Make Up', Stat/Transfer treats the first row in your file as data and assigns the field names 'col1' … 'coln'.

Lines to Skip
Enter the number of rows you want to skip at the top of the file. This option is useful for skipping headings and titles.

Read Variable Labels from Second Row
If this option is checked, variable labels will be read from the row that follows the variable names (ordinarily the second row).

Numeric Missing Value

It is possible to specify a string that will be interpreted as a missing value when Stat/Transfer reads ASCII files. For example, your input data set may use the string 'NA' to represent missing values or it may use a period.

Enter the string that represents missing values in the input data in the **Numeric Missing Value** field.

If you wish to read extended missing values for either delimited or fixed ASCII files, use the option below or enter the work "extended".

Convert extended (a-z) missing values
If this option is checked, the keyword 'extended' will be entered into the **Numeric Missing Value** field. When 'extended' is entered, extended missing values ('.a' - '.z', '.', and '._') in either delimited or fixed ASCII files will be read from the file. Note that reading missing values is case-insensitive (that is, '.a' and '.A', for example, are equivalent).

These extended missing values will be automatically written to the output file for output formats that support them (SAS and Stata).

String Quote Character
This is the character that is used to enclose string fields in the input data set. The default character is set as double quotes. You can choose the appropriate character for the input data. However, if string variables are not enclosed by any character, you can leave this option set at the default double quote.

Maximum Number of Lines to Examine
Stat/Transfer first reads your ASCII data to determine what type of variable is present in each delimited position. By default it will read the first 1000 lines of your data set. If you data are consistent, so that the first few lines suffice to show each variable type, and your data have enough rows that it actually takes more than a few seconds to examine 1000 lines, you might want to set this option to a smaller limit, such as 50. On the other hand, if you don't want to worry about this issue and you do not generally read truly large datasets (gigabytes of data), you might want to just set this to "all".

Decimal Point
If your data set uses a symbol other than the default period to indicate the decimal point in a number (a comma, for example), enter the character on the **Decimal Point** line.

Thousands Separator
If your data set uses a symbol other than the default comma to mark thousands in a number (a period, for example), enter the character on the **Thousands Separator** line.

Localize Month Names

By default, when dates are being read from ASCII files, strings in dates are compared to English month names and abbreviations. If the dates in your file use names or abbreviations in a different language and your computer's locale (i.e. date options) are set to that language, check this option and dates will be read using your language.

ASCII/Text Files - Write Options

ASCII/Text Files

Delimiter
The **Delimiter** option will give you a list of possible delimiters for output ASCII files. You can choose a delimiter from the list: commas, tabs, spaces, and semicolons, or if you have a delimiter that is not on the list, click on 'Other" and enter the delimiter you wish to use. The default is a comma.

String Quote Character
The character specified here will be written before and after string variables on output. It is typically a double quote. This character is only strictly necessary if your fields have an embedded delimiter. If you enter a blank here, string fields will not be enclosed by any character.

Numeric Missing Value
It is possible to specify the string that will be used to represent missing values when Stat/Transfer writes ASCII files. For example, you may want to create an output file for a program that expects a specific missing value, such as a period for SAS.

If you want to write a string other than the default blank to represent missing values in the output data set, then enter that string in the **Numeric Missing Value** field.

If you wish to write extended missing values for either delimited or fixed ASCII files, use the option below or enter the work "extended".

Write extended (a-z) missing values
If this option is checked, the keyword 'extended' will be entered into the **Numeric Missing Value** field. When 'extended' is entered, extended missing values ('.a' - '.z', '.', and '._') in either delimited or fixed ASCII files will be written to either delimited or fixed format ASCII files.

Line Endings
The line endings of ASCII files for Windows differ from those for Unix or OS-X. Windows files have a carriage return and a line feed at the end of each line, while Unix and Mac files have only a line feed. If you wish to write an output file for use on a Unix machine or a Mac, then you must tell Stat/Transfer to write the correct kind of line ending.

The default is 'Windows'. To write Unix and Mac files, select 'Unix & OS-X' from the drop-down menu.

Write variable names in first row

The option **Write Variable Names in First Row** is on by default. If you turn it off, field names will not be written in the first row of delimited ASCII output files.

Generated Programs and Schemas

Write complete paths

This option is used when generating programs to accompany fixed format ASCII files. When **Write complete paths** is checked, the complete path specification will be written into the programs for the output types "SAS Program + Data File", "SPSS Program + Data File" and "Stata Program + Data File". This option is checked by default.

The option **Write complete paths** is useful if you are going to read in the program and accompanying ASCII data on your own machine.

However, if you are going to store your data and program archivally or send it to another user, it is best to leave this option unchecked. In this case, only the default directory '.' and the file name are written. In this case, the program can be more easily moved to another machine, since it can be executed by setting the default directory rather than editing the program.

Shorten names and labels for older versions

If this option is left unchecked (the default) programs will be written for the latest version of SAS, SPSS and Stata, using the longest possible variable names and labels.

On the other hand, if the option **Shorten names and labels for older versions** is checked, variable names will be truncated to a width of eight characters, if necessary, and labels will be suitable for older versions of the software. Use this option if you want to maximize compatibility.

Create additional value label variables (delimited schemas)

This option is useful if your data have value labels and you want to move it to a format that does not support value labeling. An additional variable will be created for each labeled variable. These variables will have the original variable name and the suffix "_ L". Because delimited schemas are editable, you can then delete variables, re-order them and re-order them. You can then read the data back from using your edited schema and end up at the file format of your choice.

Use the number of decimal places read from input

By default, Stat/Transfer will optimize your data and write fixed format ASCII data into the narrowest width that is possible and also format ASCII data using in a manner which the decimal point is allowed to float.

In some cases, particularly for SPSS files that were originally created from fixed format ASCII data or for fixed format data read with a Stat/Transfer Schema, the input file will contain enough information to skip this optimization step and write out a file that has the same widths as the data that were originally input to SPSS. If this is the case for your data and you want fixed decimals and the original widths, check this option and also the option **Preserve numeric widths** on **General Options.**

Note that for most other file input formats, these options will result in incorrect output. You should check your results and use this option at your own risk.

SAS Reading

Stat/Transfer can read SAS user-defined formats (value labels) directly from a Windows SAS catalog file, from a catalog or data set in a CPORT library, from SAS data files for all supported platforms, and from a SAS Transport library.

SAS Value Labels - Reading
You can choose whether and how user-defined formats are to be read using the selections from the drop down menu.

Do not read formats
If this option, the default, is selected, no formats will be read.

Read directly from a catalog file (.sas7bcat)
If you have a Windows SAS catalog file containing formats and wish to read them, select **Read directly from a catalog file**.

If this option is selected, the entry '%ipath%/formats.sas7bcat' will automatically appear on the **Format Filename** line. This instructs Stat/Transfer to look for a file named *formats.sas7bdat* in the same directory as your data file.

You can change the path if your file is in a different location.

Read from a catalog in a CPORT Library (.stc)
Select this option if you have a Windows CPORT library file that contains the formats for your data set. (Your data will usually be read from a CPORT file as well, but need not be.)

If this option is selected, the entry '%ipath%/%iname.stc' will automatically appear on the **Format Filename** line. This instructs Stat/Transfer to look for a file with the same name as your input file and the extension *.stc*, in the same directory as your data file.

You can change the path if your file is in a different location.

If the **Use default catalog name** box is checked, Stat/Transfer will look in the CPORT file for a catalog named *formats*, which is the SAS default. If you would like to read your formats from a different catalog, uncheck the box, and then click on the **Read Library** button to get a list of catalogs in the CPORT file. You can then select the catalog that contains the formats that are appropriate for your data.

Read a SAS datafile (.sas7bdat)

The selection, **Read a SAS datafile**, is the appropriate option if you are working on a SAS platform other than Windows and wish to read user-defined formats. As described in the section SAS Value Labels, for cases where you are not working in Windows, Stat/Transfer can read user-defined formats that are produced in SAS data set form by PROC FORMAT using the 'cntlout' keyword.

When this option is selected, then by default, Stat/Transfer will look in the same directory as your input data set for a file named ***sas_fmts***.*ext*, where *.ext* is the extension of your input file. If you would like to use a file located somewhere else or with a different name, you can change it in the **Format Filename** edit box. You can type in a complete file specification, or you can use the macros below as part of the file specification.

%ipath% The path, including the directory, of the input file
%iname% The name, without the extension, of the input file
%iext% The extension, without the dot, of the input file

Read from a dataset in a CPORT Library (.stc)
This option is appropriate if your formats were written with the 'cntlout' option of PROC FORMAT into a CPORT library.

If this option is selected, the entry '%ipath%/%iname.stc' will automatically appear on the **Format Filename** line. This instructs Stat/Transfer to look in the same directory as your data file for a file with the same name as your input file and the extension *.stc*.

You can change the path if your file is in a different location.

If the **Use default dataset name** box is checked, Stat/Transfer will look in the CPORT file for a data set named ***sas_fmts***. If you would like to read your formats from a different data set, uncheck the box, and then click on the **Read Library** button to get list of data sets in the CPORT file. You can then check the data set that contains the formats that are appropriate for your data.

Read from a SAS Transport Library (.tpt)
This option is appropriate if your formats were written with the 'cntlout' option of PROC FORMAT into a Transport library. This is a very good way of transporting formats from an unsupported platform, such as IBM mainframe SAS.

If this option is selected, the entry '%ipath%/%iname.tpt' will automatically appear on the **Format Filename** line. This instructs Stat/Transfer to look for a file with the same name as your input file and the extension *.tpt* in the same directory as your data file. You can change the path if your file is in a different location.

If the box **Use default member names** is checked, Stat/Transfer will look in the Transport file for a member named ***sas_fmts***. If you would like to read your formats from a

different member, uncheck the box, and then click on the **Read Library** button to get a list of members in the Transport file. You can then check the member that contains the formats that are appropriate for your data.

Continue if the format file is not found

If Stat/Transfer is told to look for a user-defined format file of some type and the file containing the labels is not found when a transfer is initiated, by default the transfer will stop and an error message will be generated.

If the box **Continue if the format file is not found** is checked, processing will continue in spite of the error. Value labels will not be written to the output file.

Continue if there is an error processing formats

This will instruct Stat/Transfer to continue processing if there is an error reading the format file or if no matching formats are found in the file.

Reading SAS Extended Attributes

Extended attributes are arbitrary name/value pairs that can describe either the file as a whole or each individual variable. For instance, the attribute "level of measurement" might have values "categorical", "ordinal" and "continuous". SAS stores extended attributes in a separate file, with the extension .sas7bxcat.

Read Extended Attributes

Check this box if you would like to use the data in your SAS extended attributes file.

When this option is selected, then by default, Stat/Transfer will look in the same directory as your input data set for a file with the same name as your input file and extension .sas7bxcat. If you would like to use a file located somewhere else or with a different name, you can change it in the Attribute file name edit box. You can type in a complete file specification, or you can use the macros below as part of the file specification.

%ipath% The path, including the directory, of the input file
%iname% The name, without the extension, of the input file

SAS Writing

Write PROC FORMAT program

Stat/Transfer can optionally write a PROC FORMAT program that you can use to import value labels from other systems into SAS. To turn on this option, check the **Write a Proc Format Program** box.

The program file will by default be written to the same directory as your output data file, with the same name as your output file, but with the extension *.sas*. You can change this default in the **Filename** edit box. You can type in a complete file specification, or you can use the macros below as part of the file specification.

%opath% The path, including the directory of the output file
%oname% The name, without the extension, of the output file
%oext% The extension, without the dot, of the output file

Write SAS Catalog

Stat/Transfer can write value labels to native Windows SAS catalog files. To turn on this option, check the **Write SAS Catalog** box.

The program file will by default be written to the same directory as your output data file, with the same name as your output file, but with the extension **.sas7bcat.** You can change this default in the **Filename** edit box. You can type in a complete file specification, or you can use the macros below as part of the file specification.

%opath% The path, including the directory of the output file
%oname% The name, without the extension, of the output file
%oext% The extension, without the dot, of the output file

Note that if your catalog has the same name as your file (the default) you will need to tell SAS how to find it. Alternatively, if you name it formats.sas7bcat, SAS will find it automatically, but you should be cautious not to over-write an existing format file with the default name.

SAS Data Representation

SAS files differ depending on the platform on which they are written. Although SAS can generally read files written on platforms other than the one on which it is running, such reads are generally less efficient than reading the native format. Stat/Transfer lets

you chose from a variety of "output representations". The value for these are equivalent to those used by the SAS OUTREP= command.

You should chose the one that is most appropriate for the platform on which you are using your output file.

Date, Time and DateTime Formats

You can choose the formats that Stat/Transfer will use when writing date, time, datetime variables. You can enter a value for each of these formats. The formats will be validated when click on another tab and the widths will be set to the SAS default width for each valid format.

Write SAS Extended Attributes

If this option is checked and the input file contains extended attributes, Stat/Transfer will write them to a SAS an additional file that contains the extended attribute. This file will have the extension **.sas7xdat** and have the same name and path as the output SAS file. Extended attributes are supported by SAS version 9.4 and above.

Worksheets

You have several options to specify what part of an input worksheet to read and how to read variable names.

Data Range
You can choose different ranges to be read in input worksheets by using the drop-down menu for **Data Range**.

AutoSense This option is the default selection. When it is selected, Stat/Transfer will read to the first non-blank cell and use that as the upper left corner of the data range. By default, it will then read data until it encounters an entirely blank line. This default behavior with regard to blank rows can be changed using the **Blank Rows** options given below.

Specify Named Range   You can change the default **Autosense** behavior by specifying a named range. If you select the option **Specify Named Range**, then the **Range** line will become active and you must enter the name of a "named range" in your worksheet.

Specify Explicit Range   You can also change the default behavior by specifying an explicit range. If you select this option, then the **Range** line will become active and you must enter explicit coordinates that define the range (such as C3:F280).

When a range has been specified by either one of these methods, Stat/Transfer's default treatment of entirely blank rows will also be overridden. They will be returned in your output data and, in addition, blank rows at the end, through the last row of the specified range, will also be returned. Note that because this option generally only applies to specific worksheets and because Stat/Transfer's defaults usually work fine, the setting is not saved between sessions.

Field Name Row
By default, Stat/Transfer will use the values in the first non-blank row as the field names. You can override this in this menu. Note that the setting is not saved between sessions.

First Non-Blank Row (Default) This option will take the values in the first non-blank row as the field names even if there is no change to a numeric type in the second row.

AutoSense Stat/Transfer will attempt to autosense whether or not the data in the first non-blank row (or the first row of a specified range) are variable names or the first row of data. It does so by looking for at least one column in which there is a string in the first row and a number in the second.

No Names in Worksheet This option will treat the first non-blank row as data. Stat/Transfer will assign variable names 'col1' through 'coln'.

Specify Row Explicitly This option will take the field names from a specified row. If you select **Specify Row Explicitly** from the drop-down menu, then the **Row** line will become active and you must enter a row.

Read Variable Labels from Second Row
If this option is checked, variable labels will be read from the row immediately following the variable name row, typically the second row.

Blank Rows
This option menu is used when **Autosense** is chosen for the **Data Range** option.

Stop Reading This is the default behavior. After finding the beginning of the data by identifying the first non-blank cell, Stat/Transfer will read data until it encounters an entirely blank line. It will end the transfer at this point.

Skip Blank Rows If the option **Skip Blank Rows** is chosen, Stat/Transfer will read the entire worksheet page, searching for and returning further non-blank rows. However, any blank rows that it finds will not be written out.

Return Blank Rows If this option is selected, then all rows of the input worksheet page including blank ones will be written to the output file. This option may return unexpected blank rows from the end of a worksheet page that contains formatting but no data.

Numeric Missing Value
If there is a string in your worksheet, such as 'NA', that you would like to have treated as a numeric missing value, enter it here. Note that a single '.' is treated as missing by default.

Concatenate Worksheet Pages
The option **Concatenate Worksheet Pages** allows you to combine worksheet pages into a single output file. This option is appropriate if your worksheet contains many sheets that are identical in structure. These can be then be combined into a single output file of any type.

For example, you may have a workbook that has 50 sheets, with one sheet for each state and the same variables on each sheet. If you check this box, Stat/Transfer will effectively combine the sheets into one large input file, dropping the field names, if necessary,

on the second and higher sheets. You will end up with the data from all of your work-sheet pages in a single output file.

Transfer Blank Columns

By default, columns with variable names that have no data will be transferred. This option allows totally blank columns to be transferred as well, with the variable name set to *"coln"* where *n* is the column number.

Write A Fieldname Row in Worksheet Output

This check box controls whether or not your field names will be written as the first row of the output worksheet. By default it is checked.

Write Variable Labels to the Second Row

This option will write variable labels (if present) to the second row of the worksheet. If there are no variable labels present in the file, this option will have no effect.

Write Value Labels Instead of Numeric Values

If this option is checked, variables with value labels will be written as strings. If a value label is missing for a variable with any value labels, the numeric value will be formatted as a string and written to the worksheet.

JMP Options

Write value labels to JMP files
If this option is checked Stat/Transfer will write value labels to the output JMP file.

Use custom properties as variable labels
JMP allows users to assign custom text strings to columns. Check this box if you would like Stat/Transfer to use that text as value labels for your variables.

R and S-PLUS Options

Convert R and S-Plus factors
Both R and S-plus have a type known as a "factor". This consists of an ordered set of zero-based numbers [0…n] with a corresponding set of alphanumeric labels. For instance, a factor representing "yes" and "no" survey responses could be represented by a vector of zeros and ones and a vector of strings "yes" and "no" where 0= "yes" and 1 = "no".

Since factors have no direct analog in other statistical systems, you can choose to represent them in one of two ways.

Numeric variables with labeled values: If you choose **Numeric variables with labeled values**, Stat/Transfer will create a value label set and map it onto the numeric values. This is an appropriate choice for transferring the data into output formats that support value labels, such as SPSS, Stata, or SAS.

String variable: If you choose **String Variables**, Stat/Transfer will match the numeric variable to its string equivalent and write the string into your output dataset.

Write version one string missing
If you are using Version One of R, check this option and the appropriate value will be written for null or missing strings. If you are using a later version of R, leave this unchecked.

RATS Options

Writing
RATS options are not "sticky". They must be set before each file transfer. Also, because they may involve selecting variables from the input file, that file must be selected before they are set.

Start date and frequency
The default, **Use first date variable**, will use the first date variable in the file. By default it will autosense the frequency from that variable. If you wish to use another frequency, go to the **Frequency** box and select **Daily**, **Weekly**, **Monthly**, **Quarterly** or **Annual**.

If you have more than one date variable in your file and wish to use one other than the first, check **Use specified date variable** and then go to the **For Start Date** box and select the appropriate one from the list. If you do not wish to use the frequency default, **Autosense**, go to the **Frequency** box and select from the list.

If there are no date variables in your file, or you do not want to use them to specify the output frequency, check the **Specify a start date** option. You can then select a specific frequency from the **Frequency** list and also enter a start date in the **Select start date** box.

Output Options (1)

Stata Version
With the option **Stata Version**, you select the version you wish to use for your chosen Stata output type. You cannot select an output version lower than seven if you have chosen to write Stata/SE output.

Write Stata/SE

For Version 7 and higher, Stata comes in two flavors (Standard Stata and Stata SE or Special Edition). These used to differ in their limits for the number of output variables and the maximum permitted length of string variables. In current versions of Stata, they only differ in the limit of output variables (2047).   Stat/Transfer now defaults to Stata/SE. If you would like to enforce the lower limit on the number of variables permitted, uncheck this box.

Stata strl Threshold

Stata Version 13 and higher includes a new strL data type. These are potentially very long, variable length strings.   Stat/Transfer will write strings longer than a threshold value to Stata strl's rather than strings. The threshold, by default, is 32 characters, but you can change this here. Strings longer than Stata's width limit of 2045 characters will always be written to strls.

SPSS Version

Select your SPSS version from this list

Write Matlab Date/Time as strings

By default, Stat/Transfer will write dates, times or datetime variables into Matlab as numbers. Dates are serial date numbers with (with a base of Jan 1, 0000). Times are fractions of the day and date/time values are the sum of these two components.

You will need to use internal Matlab functions to turn these numbers into something readable. On the other hand, if you are not planning to do any computations on your dates, you can check this option and Stat/Transfer will write your date values as formatted strings in your Matlab file.

Add descriptions to value labels (/) for Statistica 7+ files

Statistica has, in addition to value labels for variables, a "description" field. If you check this option, the output value labels will be formed from the input value label plus the description, if it is available.

DDI Agency

The DDI Version 3 specification requires an "agency name" to identify the various elements in the schema. You should change the default "example.org" to something more appropriate.

Output Options (2)

These options control how Stat/Transfer constructs the names in the output section of the **Transfer** dialog from the information in the input section. Despite the setting of these options, you can always edit the names that Stat/Transfer constructs for you.

gretl Row Labels

These options are not "sticky". They must be set before each file transfer. The input file must be selected before they are set.

Use these options to control which variable, if any, is used for the row label in gretl files.

- *Use first date or string variable* uses the first date or string variable encountered in the file.
- *None* does not write a row label at all.
- *Selected* allows you to select a specific date or string variable from the **Label Variable** list below. This list will not be available until you select your input file from the **Transfer** tab.

Output table/member name rule for spaces
This option tells Stat/Transfer how to handle spaces in table or SAS member names. The default is to translate spaces into underscores.

You can choose to leave them untranslated or to have Stat/Transfer warn you if there are spaces. We do not recommend leaving spaces untranslated.

Output file name rule for invalid characters
This option tells Stat/Transfer what to do when there are invalid characters in output file names. Stat/Transfer will, by default, translate characters that are invalid in file names into underscores.

You can, instead, choose to leave them as they are, or you can have Stat/Transfer issue a warning.

Output file name Case
You can choose how you would like Stat/Transfer to handle case conversion in your file names. The default is "Smart" case conversion, in which filenames in mixed case are preserved and those in all upper are converted to lower.

In addition to this option, you can choose to have Stat/Transfer preserve the case of your file names, convert them to lower case, or convert them to upper case.

User Interface Options

Value Labels Browser

If your input file has value labels, you can display them, for each variable, on the **Variables** dialog box by checking this option. You can choose to have the value labels displayed in a **Vertical** area, just to the right of the variable names or in a **Horizontal** area, below the variable names. The default is to display the value labels in a horizontal box below the variable names.

File History Length

Stat/Transfer stores a history of the files that you have read for each **Input File Type.** You can control the number of files that are stored with this option. The default is 10 specifications for each file type.

Add Time and Date to log messages

If this is checked, the time and date will be written to each line of your log.

Clear Log before opening input file

By default, the messages that are written to the **Log** dialog box are cleared when a new file is opened. You can override this behavior and save all messages by unchecking this box. Note that if you are concerned with permanently documenting your transfer activities, you should use the **Automatic Transfer Logging** option that will automatically write a log to disk for each transfer.

Stat/Transfer GUI font

This controls the font that is used in the Stat/Transfer user interface (GUI). It will be dependent on your operating system. You can use the **Choose Font** button to call up a dialog box that will allow you to choose a different font. After you make your selection, you will need to close and then restart Stat/Transfer for your change to take effect. Note that you can get seriously ugly results by choosing an inappropriate font. It is a good idea to remember the font that you started with.

Display input and output encodings on Transfer screen

If this is checked, you will see the encoding format for both the input and output file on the **Transfer** tab.

Display Obsolete File Types

This is an option designed to reduce clutter on the input and output **File Type** drop down menus. Lotus 123 used to have 99% or the spreadsheet market, but we doubt anyone is using it today. If you would prefer not to see such old friends, check this option. But please remember that it is there, in case someone sends you a file that needs to be dusted off and converted to something more useful.

Default Directories

These options allow you to control which directories will appear by default in the input and output file selection controls on the Transfer Screen.

Input File Directory
Last Used opens the file selection control in the same directory as the last input file that was selected and transferred.

Fixed Directory allows you to choose a directory. This will be useful if your input files almost always come from the same directory.

Output File Directory
S*ame as Input* constructs an output file specification from the input file directory, the input file name, and the appropriate extension for the output file format. In addition, the drop down list under the file specification will contain a list of file specifications constructed from a most recently used list of output directories.

Last Used constructs the file specification from the last used output directory. The drop down list of other previously used paths will still be available.

Fixed Directory, will construct the output file specification from a constant directory of your own choosing.

Data Viewer Options

Long String Viewer
The long string viewer allows you to see strings that are too long to be viewed at the current column width.

Show Long String Viewer
By default, the viewer is on. To use it, simply left click on the cell you want to examine and a viewing window will display the string.   To disable it, uncheck the **Show Long String Viewer** option.

Hide Automatically
By default, the viewing window will close automatically when your cursor leaves the cell that contains the string you are viewing. To turn off this behavior, uncheck the **Hide Automatically** option.

Long String Viewer Width
The value entered here gives the maximum number of characters that will be displayed on a line of the long string viewer. If more characters are needed, the string will be wrapped in a box of the given width. If fewer are needed the size of the viewer window will be adjusted downward to fit.

Maximum Length Displayed
This is the maximum string width that will be displayed in the long string viewer. Note that the maximum length of a string in Stat/Transfer is 32767 characters.

Variable Info Viewer
If you click on the variable name at the top of the grid, a **Variable Info Viewer** box will open up that will show you the variable name, its type and, if available, its label.

Show Variable Info Viewer
The **Variable Info Viewer** box is on by default. You can turn off this feature by unchecking the box.

Hide Automatically
By default, the viewing window will close automatically when your cursor leave the cell that contains the variable you are viewing. To turn off this behavior, uncheck the **Hide Automatically** option.

Autosize Column Names
The columns in the data viewer are sized to accommodate the width of the data. By checking this box, you can have the columns sized to accommodate the variable names.

Data Viewer Font

You can use this option to control the font that is used in the data viewer. Note that if you are displaying international character sets, you should choose a Unicode font, or at the very least a font that is capable of displaying the characters that you present in your data.

Stat/Transfer Program Generation

These options control the generation of command processor programs by the Stat/Transfer user interface, whether automatically at the end of each transfer or manually, when the **Save Program** button is pressed by the user.

The programs will contain all options that are necessary to completely reproduce the transfer and are useful not only to re-run your transfer in the future, but also, together with the log files, as documentation of your work. The programs can be re-run by opening them in the **Run Program** dialog box, or by running them directly from the operating system or the command processor itself.

Automatically generate command processor program
If this option is checked a program will be generated at the conclusion of each transfer operation. This program will contain all of the settings that are necessary to reproduce your transfer operation. If you would like a program to be generated after each transfer, check this option. It is unchecked by default.

Filename
When this option is selected, then by default, Stat/Transfer will write the file to the same directory as your output file and give it the same name as your output file, but with a *.stcmd* extension. If you would like to change the location or the name, you can do so in the **Filename** edit box. Because this will be used for every transfer, you should not type in a specific name, but you can, if you wish, use a specific directory. We recommend that you use the following macros to specify the file name and also to specify a directory if you do not choose a specific one.

%ipath% The path, including the drive (on Windows), of the input data file
%iname% The name, without the extension, of the input data file
%opath% The path, including the drive (on Windows), of the output data file
%oname% The name, without the extension, of the output data file

You should always use the *.stcmd* extension.

The default filename is %opath%/%oname%.stcmd.

Ask Permission before overwriting:

<u>The program file</u>
This option controls automatic program generation. If checked (the default) you will be prompted before program files are overwritten.

The output file

This option controls the "overwrite" option that will be written to a generated program, both for automatically generated programs and those generated from the **Save Program** button on the **Transfer** dialog box.

If it is checked, the default, when the program is run, you will be asked permission before an output data set is overwritten. If it is not checked the '-y' option will be appended to the **copy** command in the program so your output data files will be overwritten without prompting.

Insert "Quit" at end of program

If this option is checked, programs run from the command processor prompt or the operating system will cause the Stat/Transfer command processor to terminate at their conclusion. It will not affect the behavior of programs run from the **Run Program** dialog box.

This option is checked by default because it allows you, for example, to click on a program in Windows Explorer to have it run and then cleanly terminate. However, you may wish to keep the command processor open to examine the results or run other programs, or execute a series of programs from another. In that case, you would not want 'quit' to be inserted at the end of your programs.

Transfer Logging / Statistics

Stat/Transfer always writes a log that is displayed in the **Log** dialog box. You can view it and, if you would like, manually save it to disk. However, the options described here control a different feature that will automatically write a log file to disk every time you do a data transfer.

Always write a log file
To turn on this feature, check the option **Always write a log file.**

Filename
By default the log file will have the same path and name as the output file, but will have the extension *.stlog.* You can change this behavior by entering a different value in the **Filename** field. You can use the following macros:

%ipath% The path, including the drive (on Windows), of the input data file
%iname% The name, without the extension, of the input data file
%opath% The path, including the drive (on Windows), of the output data file
%oname% The name, without the extension, of the output data file

If log file exists
This option controls what happens if the log file exists in the location that you have specified. The possible values are:

    Prompt before overwriting
    Overwrite without asking
    Append to existing log

The last option, "Append to existing log", is the default.

Collect anonymous usage statistics to help improve Stat/Transfer

This option controls whether anonymous usage statistics will be sent to our server. If this option is checked, the source and destination file types, number of records, number of variables and the time, are written into a file. Then, periodically, when update checks are run, these data are uploaded together with a unique code that cannot be used to identify you or your machine.

Advanced ODBC Options

User Name (UID)
Password (PSW)

On Windows, you will not need to use these options at all, since the ODBC driver manager will prompt you, if necessary.

On Linux and Macintosh, some drivers don't prompt (FileMaker is one of these), and you will need to enter your database user name and password here. For security reasons, Stat/Transfer will not store these options.

The following options are for advanced users only.

Use Unicode version of SQL API and SQL_C_WCHAR if possible

On *Windows*, this should always be checked. If it is checked, the Unicode version of the ODBC API and a wide character type will be used.

On *Mac* and on *Unix*, some older drivers do not properly support Unicode. If you are having difficulties with the Unicode default, this option should unchecked. If you uncheck it, you may have to review your driver's documentation and set the **Session Encoding** to a character set that is supported by your driver

SQL statement

These options (separately for reading and writing) allow you to enter an SQL statement that will be executed before the select statement on read and before any data are written on write. Your statement cannot return any data. For example, you can execute a Drop Table or a set commands for your database.

Restoring and Saving Options

Restore Default

The **Restore Default** button resets all of the options to their default values.

Restore Saved

The **Restore Saved** button replaces the currently selected options in the dialog boxes with those stored the last time the options were saved, either from an explicit save with the **Save** button or from your last exit from Stat/Transfer.

Save

The **Save** button saves all of the current options, with the exception of the **Write New, numeric variable name** option, the **User Missing Value** selections, and the options for **Input Worksheets: Data Range** and **Input Worksheets: Field Name Row**. This is the same behavior that occurs when you quit a Stat/Transfer session.

Advanced ASCII Options

The options control how newlines in quoted strings are handled.   To understand the options, it is first necessary to understand how properly formed ASCII, delimited files are written. First, newlines (except, obviously, those at the actual end of a line of data) should always be embedded in strings that are enclosed in double quotes. Second, if a double quote in a string, that quote should be doubled.   For example, with \n representing a newline, the following lines are properly formed.

```
"line 1 \n  line 2", "I said ""hello"""
```

Stat/Transfer will convert the embedded newline in the first string to a blank (by default), and the second string will become -

```
I said "hello".
```

Delimited ASCII files that are properly formed, such as those written by Excel , will always be read without problems. However, some software (and people), do not follow the rules.

Supposed that the second of these strings did not have escaped or doubled quotes. Stat/Transfer would terminate the string at the first embedded quote and then continue looking for the next delimiter. In the worst case it would chew up multiple lines until it found a matching quote.

The purpose of these options is to help you locate the offending line(s) in a malformed file so that you can do something about correcting them.

Newlines In Quoted Strings
The default is to "process" these. Newlines (by default) will be converted to spaces. If you select "warn" or stop, you will get a message that tells you where in your file a newline was detected. This is obviously most useful if you are not expecting embedded newlines and the problem is with quotes in your strings that are not doubled (this is a surprisingly common problem).

Newline limit per string
By default, this is "none" and, if your file is properly formed, all embedded newlines will be converted and you will have no problems. However, if there are stray quotes in your file, you should set this to zero. Then, Stat/Transfer will not chew up any additional lines and tell you the first line on which a problem was encountered.

Substitution Character

By default, newlines are converted to blanks. This will work for any destination format. However, you may wish to choose a different way of handling them. You can chose to

convert newlines to spaces, vertical bars, colons, or not convert them at all. If you chose "none" and leave them in the output, remember that newlines will cause havoc in such formats as fixed width ASCII.

Force quoted fields to strings

If this is checked, columns that contain any quoted fields will be read as string variables. This should normally be left checked, however some software will write all fields, including numeric ones with quotes around the values. If this is the case, uncheck this option.

Blank rows:
You can choose to keep blank rows, skip them, or stop the conversion.

Run Program Dialog Box

About the Run Program Dialog Box

You can now load, edit and run a Stat/Transfer command processor program directly from this tab in the user interface. Typically, these will be programs that have been automatically generated by the user interface, but you can run any Stat/Transfer program from this window.

Creating a Program Automatically from the User Interface
You can generate a program for the command processor by pressing the **Save Program** button on the **Transfer** dialog box after the transfer is complete or by setting an option to activate automatic program generation.

When this program is written, it keeps all of the selections you have entered in the dialog boxes and it translates all of the options that you have specified in the **Options** dialog box into their equivalent command processor settings. See the section Saving a Stat/Transfer Program for details.

Creating a Program Manually
You can manually create Stat/Transfer command programs as ASCII documents. Commands are entered into the document just as they would be if you were using the command processor and entering them at the Stat/Transfer prompt. See the Command Processor section for details.

Running a Program
Begin by pressing the **Open** button and selecting the program you wish to use. The program will appear in the top window of the screen, where it can be viewed and, if necessary, edited. If you make any changes to the program, the **Save** button will become active and a **Save as** dialog box will allow you to save the program back to disk.

To run the program, click on the **Run** button. Output from the command processor will appear in the **Output** window. If you want to stop a program in mid-run, click on the **Stop** button.

Log Dialog Box

About the Log Dialog Box

Click on the **Log** tab and you will see the Stat/Transfer **Log** dialog box.

This feature gives information on what is occurring during a transfer and allows better technical support.

When you carry out a transfer with the user interface, Stat/Transfer will write status, progress and error messages, which can be read as the transfer progresses and can be saved to a file. In case of errors or problems, this file can be sent to Circle Systems technical support.

Stat/Transfer Log
The log messages for a single session of Stat/Transfer will appear in the **Stat/Transfer Log** window. You can, if you wish, save these messages to a permanent file.

Log Level
This option is designed to determine the information that is written to the log window during a transfer. Currently these settings control the error messages for SAS catalog reading only. Their coverage will be expanded in the future.

The options are: 'Critical Errors Only', 'Information and Errors', 'Verbose'.

Save Log File
If you want to save the log that is written to the **Stat/Transfer log** window, click on the **Save** button.

The entry '*%opath%/%oname*.stlog' will automatically appear in the **Name** box. This instructs Stat/Transfer to write the log file to the same directory and with the same name as your output file, with the extension *.stlog*.

If you wish to use a different name, change it in the **Name** box.

The messages will then be saved to a file.

Clear
You can clear the contents of the log by pressing the **Clear Log** button.

Send Error Report

If you are experiencing a problem with Stat/Transfer, you can send an error report to our support department, by clicking **Send Error Report.**

When you click this button, another window will open, in which you can enter a complete description of the problem, your name and your email address. This message as well as your log file will be sent to Support at Circle Systems when you click the **Send the data** button.

You can, if you wish, also send the input file that is causing the problem. When you check the **Send the input file to Circle Systems** box, your input file will be sent along with the rest of the information. By default the file will be sent in a compressed and encrypted format. Including the input file will generally make it easier for Support to troubleshoot your problem.

The Command Processor

Calling the Command Processor

Stat/Transfer has a command processor that will perform transfers using a copy-like syntax and which allows "batch" jobs to be run for repetitive transfers. You can select variables and cases, automatically optimize the output variable types, or change output types manually.

You can give commands to the Stat/Transfer command processor in different ways: directly (at the Stat/Transfer prompt, or at the operating system prompt), or from a stored command file.

Starting Transfers from the Stat/Transfer Command Prompt
Starting from Windows
Go to the shortcut **Stat/Transfer Command Processor** in the **Stat/Transfer** folder under **Programs** on the **Start** menu.

The Stat/Transfer prompt will appear.

Starting from the Operating System Prompt
You can also call the command processor directly from the operating system prompt, by typing 'st'. The Stat/Transfer prompt will appear.

When you call the program, you will either need to be in the directory in which you installed Stat/Transfer, or have that directory on your path, or precede 'st' by the path to Stat/Transfer.

Starting Transfers from the Operating System Prompt
You can run a transfer directly from the operating system prompt. If you give the 'st' command and you give input and output file names, or a command file name, you will carry out an immediate data transfer, without seeing the Stat/Transfer prompt.

See:
Transfers from the Command Processor
Transfers from the Operating System Prompt

Batch Transfers with Wildcards

Wildcard Transfers
Wildcards can be used to copy multiple files in one transfer. For example:

```
copy    in/*.dta    out/*.sas7bdat
```

will convert all of the Stata files in the directory *in* to SAS files in the directory *out*.

Standard wildcards work in the input file ('?' and '*'). The output filename should always be specified with an asterisk and will be consist of the input filename, with the new extension.

We recommend that you use wildcard specifications with an input directory that contains just the files you wish to transfer. To avoid the possibility of overwriting existing files or of being prompted about this possibility, the directory should be empty on the output side. You may also specify the '-y' parameter, to suppress prompting, but you should do so with extreme caution.

Recursive Transfers

To recursively transfer all files in subfolders, use the '-r' parameter.

Transfers from the Command Processor

The basic command used when running the Stat/Transfer command processor is COPY, which has the following form:

```
    COPY    infilename.ex1    outfilename.ex2    -x1  -x2
 ...
```

where *infilename.ex1* and *outfilename.ex2* are the input and output files, respectively and *.ex1 a*nd *.ex2*, are standard extensions used to determine the type of file that is read and written. The file names can be complete file specifications.

An alternative for specifying the file type is to use a file-type tag instead of a standard extension.

Parameters *-xi* following the file names allow some options to be selected, such as selecting tables and pages, optimization of variables or controlling messages from Stat/Transfer.

You can give additional commands before you give the COPY command. These allow you to select cases and variables, to manually change output variable types, to set a number of options and to use other commands. (Note that these commands can be given only at the Stat/Transfer prompt or in a command file or in a startup file. They cannot be used when running interactively from the operating system prompt.)

If no commands for selecting cases or variables are given, then by default, all of the variables and cases will be transferred.

For example, to copy all of the variables and all of the cases from an Excel file, **indata.xls**, to a SAS file, **outdata.sas7bdat**, type:

```
    copy   indata.xls   outdata.sas7bdat
```

where the file type is given by using standard extensions.

Instead of entering commands directly, you can store them in a command file and execute that file at the Stat/Transfer prompt.

Transfers from the Operating System Prompt

You can run a transfer directly from the operating system (OS) prompt or you can execute a command file.

If you wish to run a transfer directly from the OS prompt, then the COPY command is implicit, and (if the command processor is on your path) you type the following:

```
ST  infilename.ex1 outfilename.ex2    -x1     -x2    ...
```

where *infilename.ex1* and *outfilename.ex2* are the input and output files, respectively, just as they are for the COPY command used within the command processor. File-type tags are used in the same way.

In order to run a transfer from the command line, you will either need to be in the directory in which you installed Stat/Transfer, or have that directory on your path, or precede 'st' by the path to Stat/Transfer.

The same parameters, *-xi*, that are used from within the command processor from the Stat/Transfer prompt, are also used at the OS prompt.

When a transfer is run directly from the OS prompt, variable selection and case selection are not possible. All variables and all cases will be transferred. Carrying out a transfer from the OS command line is equivalent to copying the file from one format to another.

For example, to transfer all of the variables and cases from an Excel file, *indata.xls* to a Stata file, *outdata.dta*, type:

```
st   indata.xls   outdata.dta
```

If you need to set options with SET commands before your transfer, you can do it one of two ways.

1. You can put them in a file and execute the options file by using the *-ex* switch. For instance, if your options are in a file called *myopt.stcmd*, you can execute them prior to the transfer by typing:

    ```
    st   indata.xls   outdata.dta -exmyopt.stcmd
    ```

2. If you need these options all the time, you can put them in a *startup options* file.

Wildcard transfers can be specified from the OS prompt, just as they are from the st prompt.

Combining Files

If you wish to combine multiple input files into a single output file, use the COMBINE command instead of the COPY command:

```
COMBINE  *.ex1   filename.ex2  -[f[+]filename]
```

where *.ex1* are the input files to be combined (all in the same directory), *filename.ex2* is the output file and *.ex1* and *.ex2* are standard extensions used to specify the file types. The parameter -[f[+]*filename*] is described below. The input files will be concatenated into one output file.

The syntax for this command is similar to that of the COPY command using wildcards, except that the output file specification does not contain a wildcard.

The files to be combined do not need to have identical variables and the variables do not need to be in the same order.

By default, the first file in alphabetical order becomes the "reference" file and the variables in this file and their characteristics determine the structure of the output file. The variables in succeeding files are then matched to the first file. If a file does not have a variable with a matching name or the type of the variable does not match, it will not be included in the output.

The switch '-f' allows you to specify which file will be used as the reference file. For example, if you have *file1*, *file2*, and *file3* as input and you want to use *file3* instead of *file1* as the reference, you would specify '-ffile3'. Note that you use the file name without the extension.

The file name itself may contain some information about the data in the file. For example, you might have input files with data for each state, with files named with the state name. In such cases, you may wish to have an additional variable created from the input file names and written to the output file. The parameter '-f+' tells Stat/Transfer to create a new variable from the input file names. This can be combined with the reference file specification. For example, ' -f+file3' tells Stat/Transfer to use *file3* as the reference file and to create a new output variable from the input file names.

All of the options used with the COPY command can be used with the COMBINE command.

Specifying the File Type

File Format Specification

Standard Extensions and File-Type Tags
When a COPY command is given, Stat/Transfer tries to determine the file types of both the input and output files by looking for standard extensions.

An alternative for specifying the file type is to use a "file-type tag" instead of a standard extension. In some cases, this tag is needed even when standard extensions are used.

See the table below for a list of the standard extensions and file-type tags used by Stat/Transfer. A list of standard extensions and file-type tags can be obtained by typing:

```
help formats
```

at the command prompt.

Standard Extensions
If you use standard file extensions for your input and output file names, these will usually tell the Stat/Transfer command processor what kind of file is to be read or written. Occasionally you will need to add a file-type tag as well.

File-Type Tags
File-type tags are needed in two different situations:

- For output files when the standard extension for different versions of a file type is the same.

- For file names that do not have standard extensions.

Distinguishing Versions of an Output File Type
In those cases where different versions of a given file type use the same standard extension, the standard extension will not enable Stat/Transfer to distinguish between different versions on output. To write a version other than the default version, the output file name must be preceded by a file-type tag, which identifies the appropriate output version.

Note: For JMP files, a file type tag may be necessary on input as well.

Whenever the same standard extension appears in the table below for a given file type, you must use a file-type tag. The default version is marked with an asterisk.

File Names without Standard Extensions
If your file does not have a standard extension, you must precede the file name by a file-type tag, which indicates the file type.

For example, if you want to write a Windows SAS file that has a *.dat* extension instead of an *.sas7bdat* extension, you can type

```
copy  indata.xls   sas outdata.dat
```

where 'sas' indicates that the output file is to be a SAS Windows file, while the standard extension *.xls* identifies the input file as an Excel worksheet.

Distinguishing Between File Names and File-type Tags
The command processor uses the presence of a period '.' to differentiate between extensions and file-type tags. Therefore, when using the COPY command, you must either use a period in your file name or, if the name does not contain a period, then you must use one after the file name.

For example, if you have a Stata file named *populat*, and you are going to a 1-2-3 file, *populat.wk3* , the COPY command would look like:

```
copy stata populat.    populate.wk3
```

The file-type tag must be used since the input file does not have a standard extension. The period after the file name tells Stat/Transfer that this is the file name.

The standard extensions and file-type tags used by the command processor are:

File Type	Extension	File-Type Tag	Default
1-2-3	wk?	123	
Access	mdb	access	
ASCII - Delimited	txt, csv	delim	
ASCII - Delimited with Schema	stsd (Schema file)	stdelim	
ASCII - Fixed with Schema	sts (Schema file)	stfixed	
BayseiaLab	csv	beyes	
dBASE and compatibles	dbf	xbase	
DDI Schemas	xml	ddi	
Epi Info	rec	epi	
EViews	wf1	eviews	
Excel 2007+	xlsx	excelx	
Excel 97+	xls	excel	**

Excel Version 2	xls	excel2	
FoxPro	dbf	xbase	
Gauss '96	dat	gauss	
GenStat	gwb	gwb	
HTML	htm*	html	
JMP Version *	jmp	jmp*	
Please see the discussion of JMP in the Supported Files Section.			
JSON-Stat	json	json	
LIMDEP	lpj	lpj	
Matlab 7	mat	matlab7	**
Matlab	mat	matlab	
Mineset	schema, sch	mineset	
Minitab	mtw	minitab	
Mplus	inp	mplus	
NLOGIT	lpj	nlogit	
ODBC	[none]	odbc	
OpenDocument Spreadsheet	ods	od	
OSIRIS	dict, dct	osiris	
Paradox	db	paradox	
Quattro Pro	wq?, wb?	quattro	
R	rdata	r	
RATS	rat	rats	
SAS V6 (Windows and OS/2)	sd2	sas2	
SAS V6 (Mac, Unix-HP, Sun, IBM)	ssd01	sas1	
SAS V7and V8	sas7bdat	sas	
SAS V9	sas7bdat	sas9	**
SAS CPORT	stc	cport	
SAS Transport Files	xpt, tpt	sasx	
S-PLUS	.	splus	**
S-PLUS (HP, IBM, Sun Unix)	.	splus-hl	
SPSS Data Files	sav	spss	
SPSS Data Files (HP,IBM,SunUnix)	sav	spss-hl	
SPSS Portable Files	por	spssp	
SPSS "Syntax" and Data File	sps	spsss	
SPSS Program and ASCII data	sps	spss-dat	
Stata	dta	stata	
Stata Program and ASCII Data	do	stata-dat	
Statistica Version 5 and 6	sta	statistica	**

Statistica Version 7+	sta	statistica7	
SYSTAT	sys	systat	
Triple-S	xml	sss	

Special Cases when Specifying Files

Specifying S-PLUS Files
A period must always be added to standard S-PLUS file names, since these do not have any extension.   No file-type tag is needed, however, since the standard extension for S-PLUS files is 'none'.

For example, to transfer the S-PLUS file *indata* to a Matlab file *outdata.mat*, type

```
copy indata. outdata.mat
```

The period tells Stat/Transfer that *indata* is a file name, not a file-type tag.

Specifying XML files
Several formats use "xml" as an extension. Therefore for DDI schemas, Triple-S, and XML documents, a tag must be used.

Specifying ODBC Data Sources
Note that for ODBC, the file name is not relevant. You simply use the file-type tag 'odbc' to represent the file name:

```
copy odbc outfile.sd2
```

See ODBC Data Sources for more information.

Input Worksheets, Access Data Sources, and SAS Transport Files
You can select worksheet pages, Access data source tables, and members of SAS transport files by using parameters that follow the COPY command. See Options for Pages and Tables.

Options Set by Parameters

Options for Pages and Tables

Several options are set with parameters of the form '*-xi*'. These are given after the COPY command and can be given in any order.

```
COPY   infilename.ex1   outfilename.ex2    -x1   -x2 ...
```

They can also be used at the operating system prompt:

```
ST    infilename.ex1    outfilename.ex2     -x1   -x2  …
```

Note that parameters can be entered in either upper or lower case.

Database Tables
Tables from or to databases such as Access must be specified on the command line with a parameter that follows the COPY command:

```
-Ttablename
```

For example:

```
copy dept.mdb    out.dta   -tsales
```

will move the ***sales*** table to the file ***out.dta***.

Special Considerations for Database Table Output
Sometimes, when both the input and the output file require a table or subfile specification, the '-T' switch can be ambiguous. In that case, you can modify the switch with a '>' or '<'. The parameter '-T<' means the input table specification and '-T>' means the output table specification.

 For example to transfer sheet 3 of an Excel file to an Access table named ***cities***, you could write:

```
copy  in.xls   out.mdb   -T<sheet3   -T>cities
```

Wildcard Operators
The wildcard operators '*' and '?' allow you to move more than one input Access data source table with a single command.

For example, suppose the Access data source ***company.mdb*** consists of two tables, ***sales*** and ***marketing.*** Then

```
copy   company.mdb    dept.xls    -t*
```

will move each table in ***company.mdb*** to a separate Excel worksheet. The output worksheet names will be the root name of the output file that given in the COPY command) with the table names appended, ***dept_sales.xls*** and ***dept_marketing.xls***.

Wildcards in the file name can be mixed with wildcards in the Table parameter. For example, to transfer all tables in all of the Access files in the directory ***indata*** and write them out in Stata format, you would use

```
copy   indata/*.mdb    outdata/*.dta    -t*
```

Specifying Worksheet Ranges
Specifying a Range with a Parameter
Worksheet ranges can be specified with a parameter that follows the COPY command:

```
-R[page!]coor
```

where *coor* gives the coordinates of a specific range and the optional page specification, *page*!*, gives the worksheet page, either as a name or a page number, followed by '!'. If no page is given, then by default the first page is used.

For example,

```
copy sales.wk1  dept1.sas7bdat    -ra1:i20
```

will move the range A1:I20 to the file ***dept1.sas7bdat***, while

```
copy testdat.wb3  trial1.sav    -r4!c6:t200
```

will transfer the range C6:T200 from page 4 of the worksheet ***testdat.wb3*** to the file ***trial1.sav***.

Specifying Worksheet Pages
Worksheet pages can be specified with either of the parameters:

```
-Tn              or           -Tname
```

where *n* is the page number and *name* is the page name. You must use quotes around the entire parameter if *name* contains blanks.

Examples are:

```
copy dept.xls   out.dta   -t3

copy dept.xls   out.dta   "-tnov sales"
```

<u>Wildcard Operators</u>
The wildcard operators '*' and '?' allow you to move more than one input worksheet page with a single command.

For example, suppose the worksheet *company.xls* consists of two pages, *sales* and *marketing*. Then

```
copy company.xls dept.dbf -t*
```

will move each page to a separate dBASE file. The output file names will be the root name of the output file (that given in the COPY command) with the page names appended, *dept_sales.dbf* and *dept_marketing.dbf*.

Wildcards in the file name can be mixed with wildcards in the page parameter. For example, to transfer all of the pages in all of the worksheet files in the directory *indata* and write them out in Stata format, you would use

```
copy indata/*.mdb   outdata/*.dta   -t*
```

Selecting Members of SAS CPORT or Transport Files
Whenever you select a SAS CPORT or Transport file as input, the first member will be the one used as the input data set, unless you select another one.

If the data you wish to use are in a different member, you must use the parameter '-T*membername*', where *membername* is the member you wish to select.

```
COPY   infile.xpt   outfilename.ex2   -Tmembername
```

For example, to transfer the data from the member *part4* of the SAS Transport file *indata.xpt* to the Gauss file *outdata.dat*, type:

```
copy indata.xpt outdata.dat -tpart4
```

<u>Wildcard Operators</u>
The wildcard operators '*' and '?' allow you to move more than one member of a SAS Transport file with a single command. The wildcard operators are used for SAS members in the same way that they are used for worksheet pages, as discussed above.

Options for Messages

Several options are set with parameters of the form '-*xi*'. These are given after the COPY command and can be given in any order.

```
COPY   infilename.ex1   outfilename.ex2    -x1   -x2 ...
```

They can also be used at the operating system prompt.

```
ST      infilename.ex1   outfilename.ex2    -x1   -x2 …
```

Overwrite Output File Warning
By default, Stat/Transfer will ask permission before overwriting files. If a file exists with the same name as the specified output file, you will be prompted for permission before the file is overwritten.

You can use the parameter '-y' to suppress the warning messages.

```
COPY   infilename.ex1   outfilename.ex2   -y
```

In this case, if an output file already exists with the same name, it will be overwritten with no warning.

Suppress Messages at the Stat/Transfer Prompt
When running a transfer from the Stat/Transfer prompt, by default a number of messages monitoring the progress of the transfer will appear.

The parameter '-q' will suppress all messages from Stat/Transfer, with the exception of error messages and warnings that a file is about to be overwritten.

```
COPY   infilename.ex1   outfilename.ex2   -q
```

Show Messages at the Operating System Prompt

When run from the operating system prompt, Stat/Transfer will only issue error messages and warnings that a file is about to be overwritten.

If you wish to see messages monitoring the progress of the transfer, use the parameter '-v' to get more verbose output.

```
ST      infilename.ex1   outfilename.ex2   -v
```

Selecting Cases

Cases to be transferred can be selected when you are using the COPY command from the Stat/Transfer command processor or when you are using a command file.

When you are using a command file at the OS command line, you can enter operators for case selection. However, transfers that are typed in at the OS command line will transfer all of the input cases.

You can select cases from the input file with the Stat/Transfer command processor, using the same WHERE statement that you would use with the user interface. A complete description of WHERE statements is given in the Stat/Transfer User Interface section Selecting Cases.

When using case selection with the command processor, the WHERE statement is entered before the COPY command.

The accuracy of the WHERE statement will be checked only when the COPY command is executed. For this reason, particular care should be taken when typing in variable names.

Selecting Variables

Variables to be transferred can be selected when you are using the COPY command from the Stat/Transfer command processor or when you are using a command file.

When you are using a command file at the operating system command line, you can enter operators for variable selection. However, transfers that are typed in at the operating system command line will transfer all of the input variables.

Variables can be selected with one of the commands KEEP, DROP, or TYPES. Only one of these commands can be given and it must be given prior to the COPY command.

KEEP and DROP Commands
The commands KEEP or DROP are used when you wish to select variables but do not wish to change the output variable types assigned by Stat/Transfer.

You can use the SET command DROP-KEEP to reuse the same variable list for more than one transfer.

You can give the variables that you wish to transfer or to omit either as a list or in a file.

Specifying a List of Variables
KEEP or DROP can be followed by a list of variables to be used or omitted, respectively. The variables should be separated by spaces or commas.

```
KEEP|DROP variablelist
```

Ranges can be given and wildcards are allowed. For example:

```
drop age,sat1-sat10,inc*
```

will drop the variables 'age', 'sat1' through 'sat10' and all fields beginning with 'inc'.

Using a File to Specify Variables with KEEP or DROP
If you have a file containing a list of variables to be kept or dropped, then instead of a variable list after the KEEP or DROP commands, you can specify the name of the file, preceded with an @ symbol. For example:

```
keep @varkeep.lst
```

will read the file ***varkeep.lst*** for a list of variables to be kept. These can be delimited with spaces or new lines. The VARS command, discussed below, can be used to create the file.

Note that files used with KEEP or DROP cannot contain the target output types for input variables. Such files must be read with the TYPES command.

The VARS Command
To facilitate the creation of a file containing selected input variables, a command is available that will write a list of variables to a file:

> VARS *filename variablelist* [-oc]

where *filename* is the name (with a standard extension) of the input file containing the variables to be transferred and *variablelist* is the name of the file that is to contain the data list. All of the input variables will be written, one per line, unless the parameter '-oc' has been given If variable labels are available, these will also be written as comments in the file.

The parameter '-oc', the drop-constants option, is used if you wish to drop constant or missing variables from the variable list you are creating.

Once the file has been created, you can use your favorite editor to delete the variables that are not to be used with the DROP or KEEP command.

Writing to the Screen
If, for some reason, you wish to have the input variables listed on the screen, use the VARS command without specifying an output file

> VARS *filename*

See also: Changing Output Variable Types

Changing Output Variable Types

The TYPES Command
If you wish to change the output variable target type manually, then instead of using a KEEP or DROP command, you use the TYPES command and a file, *vartypelist*, containing the selected input variables and their target output types.

```
TYPES  vartypelist
```

Note that a @ symbol does not precede the file name.

For example,

```
types type.lst
```

will read the file ***type.lst*** for a list of variables to be used, along with the target output type chosen. Each variable along with the target output type must be on a separate line.

When you use the TYPES command, you should not give any of the optimization parameters '-o'. '-of ', '-oc', or '-ofc' with the COPY command.

The VARS command, which is used to create the file, *vartypelist*, is discussed below.

TYPES CLEAR Command
When a TYPES command has been given it will be used with all succeeding COPY commands, until a new TYPES command has been given or until the following command is given.

```
TYPES   CLEAR
```


The VARS Command
The VARS command, given at the Stat/Transfer prompt with appropriate options, can be used to write a file giving the output variable types assigned by Stat/Transfer, as well as variable names and labels:

```
VARS filename vartypelist [-t] [-o[f][c]]
```

where *filename* is the name of the input file and *vartypelist* is the name of the file that is to contain the variable list with variable types. All of the input variables will be written, unless the parameter '-oc' has been given. They will be written one per line, and if variable labels are available, these will also be written as comments in the file.

The parameters '-t' and '-o' must be given. The parameter '-t' will write out the target output type for each input variable, while the parameter '-o' optimizes the output variable types. The '-of', '-oc', and '-ofc' parameters will select optimized types with the float and **drop-constants** options before writing the variable list.

Once the file has been created, you can use any editor to delete the variables that are not to be transferred and to change any output types you wish. The resulting file should still have one line for each variable that you wish to transfer.

Note that the only checking that is done at the time the TYPES command is given after a COPY command is for the existence of the file, so that editing needs to be done with care.

Changing Target Output Types
Valid types are string, byte, int, long, float, double, date, time, and date/time (see Target Output Variable Types). You should be careful not to choose a smaller type than that chosen by Stat/Transfer unless you are sure you know more about your data than Stat/Transfer does.

Output variable types can be changed freely for ASCII files and worksheet files. For all other file formats, you can change freely among the numeric types of 'byte', 'integer', 'long', 'float' and 'double' and you can change among the time types. However, do not make changes such as string to number, or number or string to date.

Remember that you are selecting a "target" type. If the output data format does not support the specific type you have selected, then Stat/Transfer will use the best match to the type you have selected

You can determine the output variable types supported by a given output file type by consulting the table given in the section describing that file format.

Using Editable Stat/Transfer Schemas

If you want to make numerous changes to your data, consider writing your data to delimited ASCII with a Stat/Transfer schema. This allows you to do many changes, such as changing variable types, renaming and rearranging variables, and so on.

See also: Selecting Variables

Setting Options with the SET Command

When using the command processor, most of the options of the user interface system are available. Some of these options are set by parameters following the COPY command. All others are selected with a SET command.

The general form of this command is:

```
SET optionname value
```

These SET commands should be given before the COPY command.

A list of available options set with the SET command can be obtained on the screen by typing either of the following:

```
help set
help set all
```

Note that these options are not available for transfers run by typing them in on the operating system command line, unless you use the startup options file.

Setting Options

If you wish, you can type each SET command in at the Stat/Transfer prompt before you give the COPY command. However, if several options are to be set, this can be tedious and error-prone. Instead, a list of all of the available options and their default settings can be written out to a file, *filename*, by typing:

```
SET LIST filename
```

Using the resulting file, settings can be conveniently edited and then used in a command file or in a startup options file.

Alternatively, automatically generated command files created with the user interface contain all of the options that are relevant to the transfer you are performing. These can be easily edited and re-run.

See: Available Options with the SET Command

Startup Options File

Profile File

You can set options globally and automatically at startup by putting them in a file called *profile.stcmd*, which should be in the same directory as your Stat/Transfer program file.

If the file *profile.stcmd* is found, it will be executed as the command processor starts up.

If you are using Windows Vista or higher, the operating system may prevent you from saving or editing a file in the *Program Files* directory tree.   There are two possible solutions for this:

1. Install or reinstall Stat/Transfer somewhere else.

2. Create a registry key that points to the directory in which your Stat/Transfer profile is located. Using Regedit, create the following key:

HKEY_LOCAL_MACHINE\SOFTWARE\Circle Systems\StatTransfer16\ProfileDir

In its value, put the full path to the directory in which your profile is located.

Running a "options file" from the OS command line

Alternatively, you can put your options into a file and execute the set commands in that file from the OS command line, along with your transfer parameter. This is done with the *-ex* switch. For example, the command:

```
 st  in.sas7bdat   out.xlsx -exmyopt.stcmd
```

will execute the commands stored in myopt.stcmd before performing the transfer.

Available Options

The following options are available for use with the SET command. Case does not matter when they are entered.

Most of these options have already been discussed in the sections on the various **Options** dialog boxes.

General Options
User Missing Values Options
Date/TimeFormats-Reading
Date/Time Formats-Writing
Encoding Options
ODBC Options
ASCII Options-Reading
ASCII Options-Write
SAS-Reading
SAS-Writing
Worksheet Options
JMP Options
R and S-Plus Options
RATS Options
Output Options
Output Options (2)
Advanced ASCII Options
Log Options

There are also options used only in the command processor.

General Options

WRITE-EXT-ATTR (Y/**N**)
Write extended attributes to output files

NUMERIC-NAMES   (Y/**N**)
Write new, numeric variable names

PRESERVE-LABEL-SETS (**Y**/N)
Preserve variable label tags and grouping where possible

SAMP-SEED   (**Auto**/*Value*)
Seed for sampling functions

VAR-CASE-CI (lower/preserve-mixed/**preserve-always**/ upper)
Variable case conversion rule for case-insensitive output file

VAR-CASE-CS (lower/ **preserve-mixed**/ preserve-always/upper)
Variable case conversion rule for case-sensitive output file

WRITE-OLD-VER   (Y/**N**) or (*Number*/**N**) **
Write older versions of output file

DROP/KEEP   (**Clear**/Save) **
Reset variable selection statement

Optimization Options

RECS-TO-OPTIMIZE   (**all**/ none/ a number)
Records to Optimize

PRESERVE-NUM-WIDTHS (Y/**N**)
Preserve the input widths of string variables if possible
PRESERVE-STR-WIDTHS (Y/**N**)
Preserve the input widths of string variables if possible

User Missing Values Options

USER-MISS (all/first/**none**)
Honor user-missing values
MAP-USER-MISS (Y/N)   -
Map user missing values to extended missing.
MAP-MISS-WITH-LABS (Y/**N**)
Map user-missing using variable labels [Y|**N**]

Date/Time Formats - Reading

DATE-FMT-READ   (**MDY***)*
Format for reading dates from ASCII files

TIME-FMT-READ   (**hms**)
Format for reading times from ASCII files

DATETIME-FMT-READ   (**MDYhms***)*
Format for reading date/time from ASCII files

CENTURY-CUTOFF   (*Valu*e)
Century cutoff for two digit years, **defaults to 30**

Date/Time Formats - Writing

DATE-FMT-WRITE   (**MDY***)*
Format for writing dates in ASCII files

TIME-FMT-WRITE   (**hms**)
Format for writing times in ASCII files

DATETIME-FMT-WRITE   (**MDYhms***)*
Format for writing date/time in ASCII files

Encoding Options

IN-ENCODING (**System**/*Other)*
Translate from this encoding when reading files

OUT-ENCODING (**System**/*Other)*
Translate from this encoding when writing files

If you choose to override the default behavior, you can choose a different encoding by going to the user interface, clicking on the **Options** tab and selecting **Encoding Options**. First select the **Region** and then the **Character Set**. The value in parentheses after the name of the character set is the value you should enter for 'Other'. Be sure to read the section Encoding Options.

ENC-ERRORS (stop/**sub**) -
On encoding errors, stop the transfer OR substitute a character. By default, it will substitute an underscore. You can specify the substitution character with the ENC-SUB-CHAR option.

ENC-SUB-CHAR (character/**underscore)**
 Substitute character for encoding errors

ENC-ERROR-LIMIT (number/ **100)** -
Maximum encoding errors before stopping

ODBC Options

ODBC-NULL-STRINGS   (Y/**N**)
Use nulls for missing strings

ODBC-USE-DATETIME   (**Y**/N)
When possible, use larger datetime type

ODBC-DATE-ROWS-TO-READ   (**10**/Number/All)
Number of rows to read to determine the type of date variable
ODBC-VARCHAR-OVER-CHAR   (**Y**/N)
Use varchar if it has the same length as char

DB-TABLE-APPEND   (Y/**N**)
Append to Access or ODBC database tables

ASCII Options-Reading

DELIMITER-RD   (**Autosense**/Comma/Tab/Space/*Other*)
Delimiter for reading ASCII files, *Other* is any character, such as '|'

COMBINE-DELIMITERS   (**Off**/Spaces only/Spaces+Tabs)
Combine adjacent delimiters when reading space delimited files

ASCII-RD-VNAMES (**Autosense**/First-row/Make-up)
Field names when reading ASCII files

ASCII-VAR-LABS-2ND-ROW (Y/**N**)
Read variable labels from row following variable names
SKIP-ROWS (**0**/Number)
Number of rows to skip at the top of the file

NUM-MISS-RD   (*Missing value characters/Extended/***None**)
Numeric missing value when reading ASCII files

QUOTE-CHAR-RD   (*Character/*None*)*
Quote character used when reading ASCII files, **defaults to "**

MAX-LINES   (*Number*/All/**1000**)
Maximum ASCII lines to examine for variable type

DECIMAL-POINT   (*Character*)
Decimal point for reading ASCII files, **defaults to '.'**

THOUSANDS-SEP   (*Character*)
Thousands separator for reading ASCII files, **defaults to ','**

ASCII Options-Write

DELIMITER-WR   (**Comma**/Tab/Space/*Other*)
Delimiter for writing ASCII files, *Other* is any character, such as '|'

QUOTE-CHAR-WR   (*Character*/None)
Quote character used when writing ASCII files, **defaults to "**

NUM-MISS-WR   (*Missing value characters/Extended/***None**)
Numeric missing value when writing ASCII files

LINE-END   (**Native**/Win/Unix)
Line endings for either Windows style or Unix style files

WRITE-FIELD-NAMES   (**Y**/N)
Write field names to ASCII files

PROG-OMIT-PATHS (Y/**N**)
Omit paths from generated program files

SCHEMA-VAR-NUMBERS (**Y**/N)
Write variable numbers in schemas for delimited files

SCHEMA-WR-LABS(Y/**N**)
Write value labels as data in schema + delimited data

USE-INPUT-DEC (Y/**N**)
Use the number of decimal places read during input.

SAS Reading

READ-SAS-FMTS   (Y/**N**)
Read value labels

READ-FMT-NAME   (**%ipath%/formats.sas7bcat**)
Name of SAS value label catalog or data file
(See below ******)
UDF-CAT-MEMBER   (**FORMATS**)
Name of the catalog in the CPORT file to be used for reading formats
(See below ******)

UDF-DAT-MEMBER   (**SAS_FMTS**)
Name of the dataset in a CPORT or Transport file to be used for reading formats
(See below ******)

CAT-ABSENCE-OK   (**Y**/N)
Continue processing if no format file is found

CAT-ERROR-OK   (**Y**/N)
Ignore errors in processing SAS catalog

** The various choices for reading SAS value labels are discussed in the section SAS Value Label Options. The following SET command sequences correspond to these choices.

Do not read formats	SET READ-SAS-FMTS N
Read directly from a catalog file	SET READ-SAS-FMTS Y SET READ-FMT-NAME *filename*.sas7bcat (the extension indicates a catalog file)
Read from a catalog in a CPORT library	SET READ-SAS-FMTS Y SET READ-FMT-NAME *filename*.stc (the extension indicates a CPORT library) SET UDF-CAT-MEMBER *catalogname*
Read from a SAS datafile	SET READ-SAS-FMTS Y SET READ-FMT-NAME *filename*.sas7bdat (the extension indicates a data file)
Read from a dataset in a CPORT library	SET READ-SAS-FMTS Y SET READ-FMT-NAME *filename.stc* (the extension indicates a CPORT library) SET UDF-DAT-MEMBER *datasetname*
Read from a SAS Trans-	SET READ-SAS-FMTS Y

port file	SET READ-FMT-NAME *filename*.tpt (the extension indicates a Transport file) SET UDF-DAT-MEMBER *datasetname*

READ-SAS-EXT-ATTR (Y/**N**)
Read Extended Attributes

READ-EXT-ATTR-NAME (filename and path/**%ipath%/%iname%.sas7bxat**)
Filename for SAS external attributes

SAS Writing

WRITE-SAS-FMTS   (Y/**N**)
Write Proc Format program for SAS

WRITE-FMT-NAME   (**%opath%/%oname%.sas**)
File name for Proc Format program

WRITE-SAS-CAT (**Y**/**N**)
Write a value labels into a SAS catalog file

SAS-DATE-FMT *(Format/***MMDDYY***)*
The SAS format used for date variables

SAS-TIME-FMT *(Format/***TIME***)*
The SAS format used for time variables

SAS-DATETIME-FMT *(Format/***DATETIME***)*
The SAS format used for DateTime variables

WRITE-SAS-ATTR (Y/**N**) -
Write extended attributes to SAS files

WRITE-SAS-CAT-NAME (**%opath%/%oname%.sas7bcat**)
Filename for SAS Catalog

WRITE-SAS-CAT-FAILURE (Y/**N**) -
Continue writing SAS dataset without value labels

WRITE-SAS-CAT-VLAB-DESCR
Write SAS catalog value label description (**Y**/N)

WRITE-FILE-LABEL (**none**/user defined)
Write SAS dataset description

SAS-OUTREP (**native**/a code)
SAS data representation

Worksheet Options
WKS-NAME-ROW (**Autosense**/1st-row/No-name/*Num*)
Reading of worksheet variable names

WKS-DATA-RANGE   (**autosense**| a named range e.g A12:Z35)
Specifying a data range as input

WKS-READ-VAR-LABS (Y/**N**)
Read variable labels from second row of worksheets

WKS-WRITE-VAR-LABS (Y/**N**)
Write variable labels into worksheets second row

WKS-BLANK-ROWS   (**Stop**/Skip/Use)
Treatment of worksheet null rows

CONCATENATE-PAGES (Y/**N**)
Concatenate worksheet pages into a single output file
WKS-NA-STRING   (**none**/*string*)
Indicate a string to be considered numeric missing when reading worksheets

WKS-RD-BLANK-COLS (Y/**N**)
Transfer blank columns from worksheets

WKS-WRITE-VNAMES (**Y**/N)
Write a field name row in worksheet output

WKS-WRITE-VAR-LABS (Y/**N**)
Write variable labels in the second row of worksheets.

XLX-WRITE-VAL-LABS (Y/**N**)
Write value labels instead of numbers to Excel.

EXCEL-NEWLINE-STR [**blanks** | a string]
Replace Excel newlines with blanks or a string.

JMP Options

JMP-LABELS   (**Y**/N)
Write value labels into JMP files

JMP-CUST-PROPERTIES (Y/**N**)
Append custom properties to variable labels when reading JMP files

R and S-Plus Options

FACTOR-AS-STRING   (Y/**N**)

Convert factors to strings (Y) or leave as numerical variables with labeled values (N)

R-OLD-NA (**Y**/N)

Write NA in R character vectors.

RATS Options

RATS-DATE-VAR (**First**, Specified, None)

The variable in the input file that specifies the series start and frequency in RATS output

RATS-DATE-VAR-NAME (Variable name)

Name of the date variable S/T has to use to determine the start date (case sensitive). If rats-date-var is set to "specified" , the variable name should be set with this option

RATS-VAR-DATE-FREQ (**Autosense**, Daily, Weekly, Monthly, Quarterly, Annual)

Frequency to be used in the output RATS file. This option will be applied if rats-date-var is set to "specified" or "first"

RATS-START-DATE (Date in DD/MM/YYYY format)

User defined start date for the output RATS time series

RATS-DATE-FREQ (**Daily**, Weekly, Monthly, Quarterly, Annual)

Frequency to be used in the output RATS file. This is applied if "rats-date-var" is set to none

Output Options

SAS-OUTREP (**native**/a code
SAS data representation

MLAB-DATETIME-AS-STRING (Y/**N**)
Write Matlab Date/Time values as strings

STAT7-VLAB-DESCRIP (Y/**N**)
Concatenate Statistica 7+ value labels and descriptions

DDI-AGENCY (*example.org*)
Agency name/url written into DDI Schema files

WRITE-STATA-SE (**Y**/N)
Adjust the limit on the number of output variables for Stata SE/MP.

STATA-STRL-THRESHOLD (number/**32**)
Strings longer than the value given in this option will be written as strl's rather than
Stata strings.

Output Options(2)

GRETL-ROW-LABEL-VAR (**first**/none/specify)

This variable will be used as the row label when writing gretl files.

GRETL-ROW-LABEL-VAR-NAME (name of string or date variable)

If gretl-row-label-var is set to *specify,* this option specifies the name of the variable. It is case sensitive.

Advanced ASCII Options

QUOTED-NL (**process**/warn/stop)
Newlines in quoted strings
QUOTED-NL-LIMIT (**none**/ a number)
Newline limit in quoted strings.
ASCII-NL-SUBST (**space**,vertbar,colon,none)
Newline substitution character.
ASCII-BLANK-ROWS (stop/skip/**use**)
Stop, skip, or use ASCII blank rows

Log Options

LOG-LEVEL**((e)rrors**,(w)arnings, (v)erbose)
Choose what will be shown in the log.

SET Options Used Only in Command Processor

Controlling the Version of Output Files

```
WRITE-OLD-VER          (Y/N) or (Number/N)
WRITE-VER (Number)
```

By default, except for Stata, a particular output file type will be written using the latest version supported by Stat/Transfer. For example SAS Version 9 files will be written if SAS is specified.

If Stata is specified as the output file type, Stata Version 16 files will be written by default. Users of older versions may see an error message, "File is not in Stata format" generated when Stata sees the newer file type. If you see this, simply select an older format and repeat your transfer.

To write a file for a previous version of a particular file type, use either of the SET commands:

```
SET WRITE-OLD-VER (Y/N) or (Number/N)
WRITE-VER (Number)
```

where *Number* is the version number of the output file that you wish to use. The number will be useful if you wish to precisely control the version of SPSS or Stata files.

If you use the first form of the SET command, Stat/Transfer will use the next-to-the-last version of the output file type you have selected. For example, if you have selected a SAS output file,

```
 set write-old-ver y
```

would tell Stat/Transfer to write a SAS Version 8 output file.

If you wished to create a Stata 5 output file, you would use

```
  set write-ver 5
```

Note that you can use either form if you wish to use the next-to-the-last version.

Reset Variable Selection Statement

```
  DROP-KEEP      (Clear/Save)
```

If data are transferred from more than one input file during a single session, then you need to specify the variables that are to be transferred from each file, using the KEEP or DROP variable selection command. The DROP-KEEP options of the SET command allow you to reuse or clear the variable selection command.

Input Files Specified Separately

If the input files are specified with separate COPY commands (that is, without using wildcards), then the default behavior is that you must give a separate KEEP or DROP variable selection statement before each COPY command. This corresponds to the option 'Clear' for DROP-KEEP.

However, if the same variables are to be transferred from each file, you can specify that the same KEEP or DROP command apply to all input files that follow, until another KEEP or DROP command is encountered. To do so use the SET command

```
set drop-keep save
```

If Stat/Transfer encounters errors in the Drop/Keep statement it will continue. You can tell it to stop by using the SET command

```
set dk-continue N
```

Input Files Specified with Wildcards
If the input files are specified with wildcards, then the default behavior is that the KEEP or DROP variable selection command you give before a COPY command applies to all of the files specified in that command. Thus the default when files are specified with wildcards is 'Save'.

If you set the DROP-KEEP option to 'Clear', then the variable selection statement will apply only to the first file. All of the other files given by the wildcard specification will transfer all of their variables.

Note that the default changes depending on whether or not the input file specification contains wildcards. The default is 'Clear' for no wildcards and 'Save' for wildcards.

Set Byte Order for S-PLUS and SPSS Output Files

```
BYTE-ORDER          (HL/LH)
```

By default, under Windows Stat/Transfer will write files in low-high byte order for S-PLUS and SPSS output files. Unix machines vary in their byte order.   DEC Alpha and

Intel processors are low-high byte order, other Unix machines are high-low. You can change the default in order to write a file appropriate for a Unix machine with the second byte order.

For example, if you wish to produce an S-PLUS file suitable for a Sun machine, you would use the SET command

```
set byte-order hl
```

Other Available Command Processor Commands

Several commands are available at the Stat/Transfer prompt which are the same as operating system commands. You can also get help from the command line and, if you wish, you can produce a log of your transfer session.

Operating System Commands

Changing Directory

You can change directories by typing the **cd** command, just as you do at the operating system prompt. Note that the **cd** command given with no parameters will return the current directory.

Changing Drive (Windows)

You can change drives by typing the drive letter followed by a colon, just as you do at the DOS prompt.

Listing Files

You can obtain a list of files by using the **dir** or **ls** command. For example,

```
dir *.xls
```

 will list all files with the extension *.xls* files in the current directory.

Running Other Programs and OS Commands

You can execute operating system commands and run other programs from the Stat/Transfer command line by preceding the program name and arguments with any of the following dos, shell or '!'. For example,

```
 shell del  logfile.dat
dos del logfile.dat
!del logfile.
```

You might use the following, allowing you to unzip a file with a command given to Stat/Transfer:

```
!unzip  sasdata.zip  mysas1.sd2
```

where '!' has been used instead of 'dos' or 'shell'*.

A sequence of commands can be entered directly at the Stat/Transfer prompt or stored in a file for batch execution.

Quit
You can terminate your Stat/Transfer session by typing 'quit' or 'q'.

Command Processor Help
Online Help
You can reach online help from the Stat/Transfer command line by typing:

```
help
```

To return to the command line, close the **HELP** screen.

Help in the Command Processor
You can issue the following commands to get help without leaving the command processor:

```
help commands
 help copy
 help formats
 help running
 help set
 help set all
 help set general
 help set date
help set sas
 help set ascii
 help set worksheet
```

To get a list of the available commands, type:

```
help help
```

Logging Stat/Transfer Sessions
You can log your transfer session to a file by typing:

```
LOG USING filename
```

where *filename* is the name of the log file.

All of the commands that you enter and all messages from Stat/Transfer will be written to the file.

Command Files

You can enter Stat/Transfer commands in a command file, as well as interactively at the Stat/Transfer prompt or the operating system prompt.

When you enter commands at the prompt, the command is executed immediately. When you store commands in a file, the commands are executed when you execute the file.

You can execute command files from the Stat/Transfer prompt or from the operating system prompt.

When you wish to run batch jobs, you must use command files.

Constructing Command Files
You can create Stat/Transfer command files as ASCII documents. Type commands in the document just as you would type them at the Stat/Transfer prompt.

You can put comments in command files by preceding the comments with a double slash, '//'.

Command File Name Extensions
Stat/Transfer assumes that command files have the extension *.stcmd* (Stat/Transfer Command).

The old extension *.stc*, which conflicts with the extension used by SAS for CPORT libraries, is still supported, but is now deprecated.

If you are executing a command file from the Stat/Transfer command line, you need not give an extension unless the file does not have the extension *.stcmd*.

If you wish to run a command file from the operating system command line, it must have the extension *.stcmd* and the extension must be given.

Executing Command Files
Executing Command Files from the Stat/Transfer Prompt
You can execute a command file at the Stat/Transfer prompt by typing:

```
EX commandfilename
```

If you enter *commandfilename* without an extension, an extension of *.stcmd* is assumed by Stat/Transfer.

A command file named **weekly.stcmd** could be executed by typing

```
ex weekly
```

Executing Command Files from the Operating System Prompt

You can execute a command file at the operating system prompt by typing

```
ST      commandfilename.stcmd
```

In order to use command files from the operating system prompt, you will need to be in the directory in which you installed Stat/Transfer, or have a path defined to it or give the path explicitly.

For example, a command file, ***repeat.stcmd***, will be executed if, at the operating system prompt, you type:

```
st    repeat.stcmd
```

Note that the extension, which is explicitly given, must be ***.stcmd***. If the file is not in the Stat/Transfer program directory, the complete path to it must be given.

If you do the same transfers all of the time, the ability to execute command files in this way allows you to set up a shortcut that will perform your transfers with a single click on a desktop icon.

Running and Editing Command Files from the Windows Explorer

When you double click on a Stat/Transfer command file in the **Windows Explorer**, the command processor will be launched and the command file executed.

If the last command in the file is 'quit', you will be taken directly back to the **Explorer**. Otherwise, the command 'quit' at the command line will return you to the **Explorer**.

You can edit a command file by right-clicking then selecting 'edit'.

Running Programs and Commands in Command Files

Command files can contain commands to execute programs, as well as Stat/Transfer commands.

These options will, for example, allow you to zip or unzip files with commands given to the Stat/Transfer command processor. For example, a command file might contain the following set of commands.

```
// unzip a SAS file
!pkunzip  sasdata.zip  mysas1.sd2

// copy it to an SPSS file
copy  mysas1.sd2  myspss1.sav

// zip your new SPSS file
!pkzip  myspss1.zip  myspss1.sav

// delete the SPSS file
!del myspss1.sav

// delete the SAS file
!del mysas1.sd2
```

where '//' indicates to the command processor that what follows is a comment and '!' has been used instead of 'dos'.

A sequence of command such as these can be entered directly at the Stat/Transfer command prompt or stored in a command file for batch execution.

ODBC Data Sources

The DBR and DBW Commands

Generally, the process of opening an ODBC data source for reading or writing requires information beyond a simple file name. Thus, you may need to give commands which define the data source name, the table name, and sometimes a password or database name.

Stat Transfer has two commands specifically for ODBC data sources:

    DBR (short for database read)
    DBW (short for database write)

These commands must precede the COPY command.

Each of these commands is followed by a single parameter that gives necessary information for the ODBC data source. Each of these parameters consists of the parameter name followed by a value for that parameter.

```
    DBR | DBW parametername  value
```

The parameter names are:

DATASOURCE - the name of the ODBC data source
DATABASE - the name of the database
PASSWORD - logon password for the data source
TABLE - the name of the table you want to read or write
SERVER - the name of the machine the database is running on
SQL - a sql statement to retrieve specific fields and/or join tables
CONNSTR - an ODBC connection string (explained below).

The parameter CONNSTR is the most useful of these in command files.

Which of the parameters is necessary for a particular database depends on the database and on the driver. You must enter as many DBR and DBW commands as necessary to supply all of the parameters.

For example, if you want to read from the **employee** table of a SQL Server database you could specify the table with the following command.

```
    dbr table employee
```

You could then submit a COPY command:

```
copy ODBC out.sd2
```

to copy the table to a SAS file named ***out.sd2***.

To write to a table, ***data***, in an ODBC data source, you could use the commands:

```
  dbw table data
copy indata.sd2 ODBC
```

Retrieving Parameter Values
During a Stat/Transfer session, you can obtain the value of any parameter by entering DBR or DBW and the parameter name without a value:

```
DBR | DBW parametername
```

If no value has been entered yet, you will be told that the value is blank.

Clearing Parameter Values
The DBR and DBW values are persistent across COPY commands during a session. Thus, you will need to change them or clear them if you are executing multiple COPY commands using different data sources.

You can clear the value of any parameter by entering 'clear' in place of a value. For example:

```
dbr connstr clear
```

clears the value of the parameter CONNSTR. Note that it is necessary to 'clear' parameter values if you want to be re-prompted by the ODBC driver manager for any values that you have previously entered.

Running Batch Jobs with ODBC

With SQL Server and other client-server databases, you are likely to need a password and other parameters. Fortunately, when you connect from the command line, the ODBC driver manager will prompt you for whatever information is needed. However, in order to run a job in an automatic batch mode with a command file, you need to supply the necessary information automatically.

The simplest way to do this is to take advantage of the fact that ODBC will return a connection string that contains all of the information that you entered. This information is in the proper form to complete the connection again without any prompting from the ODBC driver manager.

You can obtain the connection string from an ODBC data source by connecting interactively to the source and then entering either:

```
dbr connstr
```
or
```
dbw connstr
```

You will find that the connection strings are generally long and complicated, so that typing one into a command file after retrieval is tedious and error prone. Furthermore, because the connection string may contain your password, you might not want it written openly to a command file.

Stat/Transfer solves these difficulties with operators that allow you to write or retrieve the value of any parameter and by allowing encryption.

The Write and Retrieve Operators
The operator '>' allows you to store the value of any parameter in a file, while the operator '<' allows you to read back a stored value.

```
DBR | DBW  parametername   > | <   storagefile
```

Thus, in order to write the connection string for an input ODBC data source to a file, **connect.str**, type:

```
dbr connstr >  connect.str
```

To retrieve the connection string and make it available to an input ODBC data source, type:

```
dbr connstr <  connect.str
```

To use a connection string in a batch job, you can connect interactively to the ODBC data source, use the '>' operator to store the connection string in a file and then use the '<' operator with a DBR or DBW command in a command file.

Encrypting Connection Strings
Because the connection string may contain your password, you may not want it written to unprotected files. You can encrypt and decrypt the stored value of any parameter by adding a pound sign to the '>' or '<' operators.

For example,

```
dbr connstr >#  connect.str
```

writes the connection string in encrypted format and

```
dbr connstr <#  connect.str
```

retrieves and decrypts the connection string.

Variable Naming and Limits

Variable Names

Stat/Transfer will, by default, ensure that variable names are legal in the target data set and are unique. It does this by adding special characters when needed, changing case where appropriate, substituting underscores (or another legal character) for invalid characters and by truncating variable names to an acceptable length. It then eliminates duplicates created by this process by appending a number to the end of the second and higher instances of duplicate variable names.

The original variable names will, if possible, be retained as variable labels.

If you check the option **Write new, numeric variable names. (VN)**, in the **General Options** section of the **Options** dialog box, then instead of the default variable names, Stat/Transfer will create new variable names of the form V1...VN. This is chiefly useful when dealing with truncated names, which often have little resemblance to the names you started with. If your output system supports variable labels, you may wish to use this option to assign numeric names for your variables. You can then use the variable labels for description.

Because this option is likely to be useful only in special circumstances, it reverts to the default between sessions of Stat/Transfer.

Limitations on the Number of Variables

Stat/Transfer will check whether the number of variables that have been selected exceeds the maximum permitted by the target program and will inform you if too many variables have been selected.

Limitations on Strings with Value Labels

Both SAS and SPSS support the labeling of string variables. Stat/Transfer will auto-matically transfer such value labels, both to the internal file formats of SAS or SPSS and to the program files written by Stat/Transfer to create fixed format ASCII files.

Internal Limitations

The maximum width of an alphanumeric variable is 32,000 characters.

Variable names are limited internally to a maximum length of 255 characters.

Data are generally processed one case at a time, so the number of cases is not limited. Exceptions to this are the worksheets and file formats that must be transposed, such as R. These transfers are limited by available virtual memory. It is unlikely, however, to be a problem.

Return Transfers to the Original Format

Because systems vary widely in the types of variables they support and the information they store about them, you must be careful when transferring data from one format to another and then transferring back to the original format. Stat/Transfer will always transfer numeric values without losing accuracy. However, other information about variables may be lost.

For example, SPSS maintains value labels and up to three types of missing value for each variable. If you transfer from an SPSS data set to Access, the value labels will be dropped and missing values will be translated into a single Access internal format. If you were then to transfer your data back to SPSS, missing numbers would all be translated into the SPSS system missing value, not the original values, and because Access-does not store value labels in the data set, the value labels would be lost.

Because many systems do not explicitly support dates, you should be particularly cautious when planning two-way transfers of date variables. Stat/Transfer will generally convert date variables into MM/DD/YY strings when transferring to systems that do not explicitly support dates. This makes the date intelligible in the destination system. However, when you transfer back to the source system, Stat/Transfer has no way of knowing that these strings are dates and will treat them simply as character variables.

If you are planning such a two-way transfer, you may wish to treat date data as numbers rather than as dates. Stat/Transfer stores dates internally as the number of days since December 30, 1899. Thus, if you wish, you can choose to put dates directly into your destination data set as numbers in this form by selecting the proper target type at the Variable Selection dialog box. If you do so, these will survive a two-way transfer involving a system that does not explicitly support dates.

Supported Programs

Input and Output Variable Types

For a given input file format, Stat/Transfer will select a target output variable type for each input variable type. For any given input file, these target types are automatically chosen during the optimization pass during a transfer. If you wish to see the target types that have been chosen, use the Optimize button in the **Variables** dialog box or look in the file generated by the **VARS** command. These target output types can be changed manually.

The actual output variable type that results from each target variable type will depend on the output file format. The available output variable types are given for each of the supported file formats in the section for that specific format.

Note that all numerical variables are stored internally in Stat/Transfer as double precision floating point numbers and are then written out according to the assigned variable type. Thus, no numerical precision is lost.

1-2-3 Worksheet Files

Stat/Transfer will read and write files from any version of Lotus 1-2-3 (including Release 3.x and Windows).

***Standard extensions*:**

Lotus 1-2-3 Version 1 through 2.x	**wk1**
Lotus 1-2-3 Version 3.x and Windows	**wk3**

Database Format

Worksheet database files are structured worksheets where each row is a single case and each column contains a variable.   Data can consist of numbers (including serial date numbers), labels, or formulas.

The first non-blank row of a worksheet database file usually has strings in each column that give the names of the variables. The data then begins in the next row. However, variable names may be in different rows or not present at all.

You have several options to specify what part of an input worksheet to read and how to read variable names. An option allows you to read variable labels from the row after the variable names.

Data Range

You can choose different ranges to be read in input worksheets by using the drop-down menu for **Data Range** in the **Worksheets** section of the **Options** dialog box or using the SET command, WKS-DATA-RANGE.

If you use **Autosense** (the default), Stat/Transfer will read to the first non-blank cell and use that as the upper left corner of the data range. It will then read data until it encounters an entirely blank line. This is the default behavior. You can change the behavior with respect to blank rows by using the **Blank Rows** option.

Rather than have Stat/Transfer automatically sense the number of rows to be read, you can use the other options for **Data Range** to specify a range, either by giving a named range or by giving explicit coordinates.

When a range has been specified, Stat/Transfer's treatment of entirely blank rows will also be overridden. They will be returned in your output data and, in addition, blank rows at the end of the worksheet, through the last row of the specified range, will also be returned.

Determining Variable Names

By default, Stat/Transfer will attempt to determine whether or not the data in the first non-blank row (or the first row of a specified range) are variable names or the first row of data. It does so by looking for at least one column in which there is a string in the first row and a number in the second.

If this behavior is inappropriate for your worksheet (for example, if you have only string data), you can specify different options in the **Field Name Row** drop-down menu of the **Worksheets** section of the **Options** dialog box. You can specify that variable names must be taken from the first non-blank row, that they be taken from a specific row or that the worksheet does not contain variable names, so that Stat/Transfer should assign them ('col1' through 'col*n'.*)

Determining the Data Types and Widths

After identifying the label row, Stat/Transfer will look at the entire column. If any cell contains a string, the entire column will be treated as string data (with numbers and dates converted to strings). If the column is all numeric or blank, it will be treated as a numeric column.

The width of the column for each numeric variable and the format of the first non-blank data cell in that column are used, where possible, to set the default target, or output, types for the numeric variables. If the format of the first data row has any decimal places (for example, F(2)), the target type will be 'float'. On the other hand, if the cell format has no decimal places (for example, F(0)) the target types will be various flavors of integers which depend on the column width. If the column width is less than three, the target type will be 'byte'. If the column width is less than five, the target type will by 'int'. Otherwise the target type will be 'long'. Any date format in the first data row will set the target type to 'date'.

The maximum width of character variables is determined by examining the widths of all of the strings in a column.

Stat/Transfer is lenient in typing variables from worksheets. If it is expecting a character variable and it encounters a number it will convert it to a string.

Combining Multiple Input Worksheet to a Single Output File

The option **Concatenate Worksheet Pages**, found in the **Worksheets** section of the **Options** dialog box allows you to combine worksheet pages into a single output file. This option is appropriate if your worksheet contains many sheets that are identical in structure. These can be then be combined into a single output file of any type.

For example, you may have a workbook that has 50 sheets, with one sheet for each state and the same variables on each sheet. If you check this box, Stat/Transfer will

effectively combine the sheets into one large input file, dropping the field names, if necessary, on the second and higher sheets. You will end up with the data from all of your worksheet pages in a single output file.

Writing 1-2-3 Worksheet Files
On output, Stat/Transfer will write variable labels in the first row of the worksheet. Data values will be placed in the second and succeeding rows.

Column widths and formats will be determined by the variable information available. Dates and character variables are straightforward. For numerical data, information on the width and number of decimal places of variables, where available, is used to set the column widths and formats.

Missing Data
On input, blank cells and cells containing labels consisting of a single dot are read as missing.

If there is a string in your worksheet, such as 'NA', that you would like to have treated as a numeric missing value, you can specify it using the **Numeric Missing Value** option in the **Worksheets** section of the **Options** dialog box.

When transferring data to worksheets from other formats, missing values will be written out to worksheets as blank cells.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float double	Numeric cell (formatted if information available)
string	Label
date	Serial date number (with date format)
time	Time fraction (with date format if available)

date/time	Date number + time fraction (with date/time format if available)

Access

Stat/Transfer will read and write Access databases on Windows.

Although the data are transferred through the Microsoft Access ODBC driver, Access files are treated as a normal Stat/Transfer file type. You can thus choose the file from the normal **Open** or **Save** dialog boxes and need not be concerned with the process of configuring an ODBC data source for each file.

In order to use Microsoft Access you must have the ODBC and Access components installed with the Stat/Transfer program. If you did not do this when you installed Stat/Transfer, you can re-run the installation program and install these components.

Standard extension: **mdb**

Reading Microsoft Access Databases
Stat/Transfer can either read single tables or multiple tables that are joined in an Access "view".

Access has a single "date" type 'Date/Time', which is used to store date, time and date/-time values. The format of the column determines how these are displayed. There is no straightforward way outside of Access to determine how a column is formatted. By default, Stat/Transfer reads the first ten cases of the file to determine if each 'Date/Time' column contains date data, time data or both. This information is used to set the type. You can change this default in the **ODBC/Access Options** of the **Options** dialog box.

Variable labels are now transferred when reading Access tables.

Writing Microsoft Access Databases
On output, tables can be created in a new file, new tables can be created in an existing file, or existing tables can be overwritten with a new table.

In addition, new data can be now appended to an existing database table. This option is off by default and must be turned on by using the **Append to Access and ODBC Tables** option in the **ODBC/Access Options** of the **Options** dialog box or the SET command DB-TABLE-APPEND in the command processor.

Stat/Transfer will match as many variables as possible to those already in the table and add your data to the matching columns. Obviously at least one column must match exactly and, in addition, the table must be free of constraints, such as those requiring unique keys, that would prohibit a simple 'append' operation.

When writing time or date data to Access, there is no way for Stat/Transfer to set the format. Because of this, these variable types are written as 'Date/Time' values. You may, if you wish, change the format with Access after you have transferred your data.

Missing Data
Access supports a null value, which is the same as Stat/Transfer's missing value.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int	Short
long	Long
float	Single
double	Double
date time date/time	Date/Time
string	Character

ASCII Files - Delimited

Stat/Transfer will read and write delimited ASCII. Note that as of version 10, "ASCII" is used as a shorthand, since Stat/Transfer supports a variety of character sets besides ASCII. If you need to read or write data in a character set that is something other than your system's default character set, you will need to set the encoding options.

Standard extension: **txt. csv**

Reading ASCII Delimited Files
When reading ASCII delimited files, you can use the user interface or command processor options to specify:

- What character is read as the delimiter. You can explicitly specify the character or you can allow the program to sense it automatically.

- That Stat/Transfer should combine adjacent blank delimiters.

- Whether or not the first row is treated as variable names, or whether Stat/Transfer should automatically sense it or whether Stat/Transfer should assign field names.

- The character that is used to enclose string fields in the input data.

- The number of lines Stat/Transfer will read to determine each type of variable present. By default the entire data set is read.

- What character is read as missing value.

- The decimal point and thousands separator in your data.

- The user-defined DateTime format is used to determine whether a given field is a date, time or a date/time. The formats for dates, times and date/times are then used to actually read the data. If your dates are not in the default MDY format, you will need to change the formats in the option settings
- A century changeover year. If you are reading two-digit years, you can use this option to control how they are read. The default for the option is '30'.

All of the user interface options available for reading ASCII files are set in the **Options** dialog box, in the sections Date/Time Formats - Reading and ASCII/Text File - Read Options. The command processor options can be found in Setting Options with the Set Command.

If you wish to have more flexibility in reading delimited files, you can describe the file with a Schema file.

Writing ASCII Files
When writing ASCII delimited files, you can specify:

- The character that will be used as the delimiter in each record: commas, tabs, spaces, semi-colons, or some other character.

- The character that will be used to enclose string variables on output. It is typically a double quote.

- What character will be used for missing values.

- What line ending to use.

- The date and time formats.

- Whether Stat/Transfer should write variable names in the first row of the output.

All of the options available for writing ASCII files are set in the **Options** dialog box, in the sections Date/Time Formats - Writing and ASCII/Text File - Write Options. The command processor options can be found in Setting Options with the Set Command.

Missing Values
By default, missing values are indicated on input and output by one delimiter imme-diately following another. You may change this default behavior in the **ASCII/Text File - Write Options** section of the **Options** dialog box. Extended missing values are supported.

When an ASCII file with extended missing values is transferred to a SAS or Stata file, the input missing values will transfer to the equivalent SAS ones. When a SAS file is transferred to an ASCII file with extended missing values specified, any missing values '._' in the input SAS file are written out as '.' in the output.

Note that if a blank is used as the delimiter, missing values will be hard to determine.

Output Variable Types
The output variable type that results from each target variable type is given in the fol-lowing table:

Target Type	Output Type
byte int long float double	Number (with a precision of up to 15 decimal places)
string	Character
date time date/time	Character (written using ASCII format options currently in effect)

See also:
ASCII Schema Files

ASCII Files - Fixed Format

Stat/Transfer will read and write fixed format ASCII.

***Standard extension*: txt. csv**

Reading Fixed Format ASCII Files
Stat/Transfer can read fixed format ASCII files. These are data files where each field has a fixed column location and width, rather than having the start and end of the field marked by a delimiter character, as in delimited ASCII files.

Unlike the delimited ASCII format, where it is possible for Stat/Transfer to read the data based on the information contained in the data file, fixed format files require a separate description of each field's location in the record. You will need to provide this inform-ation in a separate Stat/Transfer Schema file. In the Schema file you can also define missing values, value and variable labels, and other characteristics of your data, as well as the path to the data file or the data file itself.

Using the User Interface
To use fixed format ASCII files as input into Stat/Transfer using the Stat/Transfer user interface, select "ASCII - Fixed Format (S/T Schema)" from the list of supported file types at the **Input File Type** line of the **Transfer** dialog box.

In the **File Specification** line of the **Transfer** dialog box, rather than entering the name of the fixed format ASCII file itself, enter the name of the Schema file that describes it. Schema files will generally have the name of the input fixed format data file and the extension *.sts*. The Schema file then points to the data file. Thus, the location of the data file must be accurately given in the Schema file. If Stat/Transfer wrote the Schema file originally, then the exact path that Stat/Transfer used to write the ASCII file on your machine was entered as the file location in the Schema file.

The file location can be changed by opening up the Schema in any text editor, such as notepad, and giving the file location with the FILE command. Comments in the first lines of the Schema or program file will guide you in changing the specified file loc-ation.

Using the Command Processor
To specify a fixed format ASCII file as input to Stat/Transfer when using the command processor, give the name of the Schema file as the input file in the COPY command. The file should have the extension *.sts*. If for some reason your file does not have the extension *.sts*, then you can use the file-type tag ***stfixed***.

The Schema file then points to the data file as described above.

Writing Fixed Format ASCII Data Files
On output, Stat/Transfer can write fixed format data files, and will, if directed, create a Stat/Transfer Schema file at the same time. Alternatively, you can choose to have Stat/Transfer write programs to accompany output fixed format ASCII files that will allow these files to be read into SAS, SPSS, or Stata.

Fixed format ASCII data files, along with the associated program files, are useful for passing your data to colleagues. In addition, they are a prudent way to archive your data for future use, as they are not dependent on changes in computer architecture or the fate of a particular software package.

You have various options for writing out the program files. If you are using the user interface, see ASCII/Text File Write and Date/Time Formats-Writing in the **Options** dialog box. For the command processor, see Available Options with the SET command.

Using the User Interface
When you click on the **Output File Type** control in the **Transfer** dialog box, you will see the following entries in the list of supported output file types:

ASCII/text - Fixed Format (S/T Schema)
    …
SAS Program + ASCII Data File
    …
SPSS Program + ASCII Data File
    …
Stata Program + ASCII Data File

ASCII/text - Fixed Format (S/T Schema)
This selection will write an ASCII data file plus a Stat/Transfer Schema file.

The Schema file name will appear in the output **File Specification** line of the **Transfer** dialog box. Stat/Transfer constructs the output file specification by using the same drive (Windows), directory and name as the input file and appending the file extension *.sts*. The output file itself will have the same name as the input and Schema files, but with the extension *.dat*. The complete name and path of the output ASCII data file will be encoded in the Schema file.

The data file and Schema file can be read back by Stat/Transfer. This is useful, for instance, if you want to add variable labels, value labels, and/or missing values to data stored in a file format that does not support these features.

For instance you could write the contents of an Excel spreadsheet to a fixed format ASCII file and then edit the contents of the Schema to add missing values or value labels that could then be transferred to SPSS, SAS or Stata.

XXX Program + ASCII Data File
The last three selections above

    SAS Program + ASCII Data File
    SPSS Program + ASCII Data File
    Stata Program + ASCII Data File

will each write a fixed format ASCII file along with a single program that will allow the file to be read into SAS, SPSS or Stata. In addition, for the selection "Stata Program + ASCII Data File," a dictionary file will be written for Stata.

The program file name will appear in the output **File Specification** line of the **Transfer** dialog box. Stat/Transfer constructs the output file specification by using the same drive (Windows), directory and name as the input file and appending the file extensions *.sas*, *.sps*, or *.do*. The output file itself will have the same name as the input and program files, but with the extension *.dat*. If a Stata program file has been selected, the dictionary file will have the same name as the input and program files, but with the extension *.dct*. The complete name and path of the output ASCII data file will be encoded in the program file.

You can then load the program file (that with the *.sas*, *.sps*, or *.do* extension) directly in SAS, SPSS or Stata to produce a data set with same name as the program file. If necessary, the ASCII file specification encoded in the program files can be changed using any text editor that handles plain text (see below).

Using the Command Processor
You can write fixed format ASCII files using the command processor.

The only change from the description given above for using the user interface is that the type of output is specified by using an appropriate extension for the output file name in the COPY command, rather than making a selection from the dialog box.

The file extensions used to specify each type of output are:

Fixed format ASCII with a Schema file	.sts
Fixed format ASCII with a SAS program	.sas
Fixed format ASCII with an SPSS program	.sps
Fixed format ASCII with a Stata program	.do

Reading Program Files + ASCII Data Files

Stat Transfer can read fixed format data from SPSS and SAS programs. (Stata is read only) In general, we can read back programs that we have written. But the languages are very big and not every construct is supported. The supported keywords are those that are most commonly used for describing publicly distributed data. We do not support any statements that compute new values and variables.

If Stat/Transfer encounters a construct that it does not understand, a message will be written into the log and the job will continue. You should set the log level at **Information and Errors** or higher, and always examine the log

Keywords supported for SPSS:	Keywords Supported for SAS
FILE	FILENAME
DATA LIST	INPUT
BEGIN DATA	PROC FORMAT
FORMATS	VALUE
VALUE LABELS	LABEL
VARIABLE LABELS	FORMAT
MISSING VALUES	IF (when used for recoding into user-missing values)

Changing File Specifications in the Schema or Program File
When a fixed format ASCII file is used as input, either to Stat/Transfer with a Schema file or to SAS, SPSS or Stata with a program file, the first file that is read is either the Schema file or a program file. This file then points to the actual data file. Thus the location of the data file must be accurately given in the Schema or program file.

When the ASCII file with its accompanying Schema or program file is written on output, by default, the exact path that Stat/Transfer uses to write the output ASCII file on your machine will be entered as the file location in the Schema or program file. If you change this behavior with the options for **Generated Programs and Schemas** in the **ASCII/Text Write** section of the **Options** dialog box, the output ASCII data file specification will be written to all of the program files using the directory "./" as the path.

The file location can be changed by opening up the Schema or program file in any text editor, such as notepad and giving the file location with the FILE command. Comments in the first lines of the Schema or program file will guide you in changing the specified file location.

Missing Values

By default, missing values are indicated on input and output by a blank. This behavior can be changed by setting the **Numeric Missing Value** option to "extended" in the **ASCII Read** or **ASCII Write** sections of the **Options** dialog box.

When an ASCII file with extended missing values is transferred to a SAS or a Stata file, the input missing values will transfer to the equivalent SAS ones. When a SAS file is transferred to an ASCII file with extended missing values specified, any missing values '._' in the input SAS file are written out as '.' in the output.

Output Variable Types

The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float double	Number (with a precision of up to 15 decimal places)
string	Character
date time date/time	Character

See also:
[Schema Files for ASCII Input](#)

BayesiaLab

BayesiaLab provides an environment for machine learning, knowledge modeling, analytics, simulation, and optimization — based on the Bayesian network paradigm. See http://www.bayesia.com/   Stat/Transfer provides support for writing BeyesiaLab files.

Standard extension: *csv (the tag **beyes** must be used in the command processor)*

Writing Epi Info Files
For a dataset that has both value and and variable labels, four files will be produced. The first of these is a delimited ASCII file with the data. This has the extension *.csv*

The other files are as follows:

NodeComments.txt (A mapping of short variable names to variable labels)
ClassNames.txt (A mapping of short variable names value label names or "tags"
StatNames.txt (These are the actual value labels along with their tags)

These files will only be written if they are necessary. For instance, if the input file does not have any value labels, the final two files will not be written.

These files can then be loaded into BayesiaLab and then saved as a native file.

Missing Data
Missing values will be written as a period.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float double	Number (with a precision of up to 15 decimal places)
string	Character
date time	Character (written using ASCII format options currently in effect)

date/time	

SCHEMA Files for ASCII Input

The Schema file is used to describe an input ASCII file to Stat/Transfer.

If you are using a fixed format ASCII file that was not created by Stat/Transfer, then you must create the Schema file before you read a fixed format ASCII file into Stat/Transfer.

If you are using a delimited ASCII file as input, you may use a Schema file to specify variable names, labels and missing values in a more detailed way than is allowed by the automatic method described in the section Delimited ASCII Files.

The Schema file should be a plain ASCII text file with the extension *.sts* (Stat/Transfer Schema) or *.stsd* .(Stat/Transfer Schema - Delimited). You can create it with any text editor that will create plain text.

It is most straightforward if your ASCII data file has the same name as the Schema file, with the extension *.dat*, and is located in the same directory as the Schema file. If this is not the case, you can override this default with the FILE command, as discussed below.

Note that the Schema syntax is similar to SPSS DATA LIST syntax, but is more convenient to use. There is no MISSING VALUES command, because missing values can be defined along with the variables. Similarly, variable labels can be defined alongside the variables. Value labels are attached with tags in a manner that is closer to SAS and Stata than SPSS.

Creating a Schema File
The structure of the Schema file is the following:

```
FILE   file specification            when required
FORMAT [fixed] | delimited  delimiter
FIRST LINE  n           when required
VARIABLES
       /           when required
       Variable name | variable list | variable range
columns  (variable type)     Fixed
       Variable name [variable number]  (variable type)
Delimited
       Optional elements for the VARIABLES command
               [ Missing value specs ]
               { Variable label } }
               \ Value label tag
Optional commands
VARIABLE LABELS
```

```
      Variable name    variable label
VALUE  LABELS
        \label tag
              value   label
                    …
              value   label
DATA              when required
```

Each of the major commands (FILE, FORMAT, FIRST LINE, VARIABLES, VARIABLE LABELS, VALUE LABELS, and DATA) must begin in the first space on a line. All other elements must be indented at least one space or tab.

Commands in the Schema file are not case sensitive.

For fixed format data, the most basic Schema file consists of a list of variable names, followed by their column locations and, for non-numeric variables, the variable type in parentheses.

For delimited data, the most basic Schema file consists of a list of variable names and the types, in parentheses, of all variables.

All of the elements of a Schema file are described below.

FILE Command
If your ASCII input file is not in the same directory as your Schema file, if it has an extension other than *.dat*, or has a name that is different from the Schema file, you must use the FILE command.

This consists of the word FILE beginning in the first column, followed by the complete path and name of your input data file. If there are embedded spaces in any of these elements, you must enclose the file specification in single or double quotes.

For example:

```
     File "/my data files/new/big.file".
```

If the ASCII input file is in the same directory as the Schema file, with the same name and with an extension *.dat*, then the FILE command does not need to be used.

The FILE command is also omitted if you choose to put your input data in the same file as the Schema commands, using the DATA command (see below).

Note that when Stat/Transfer writes a Schema file for an output fixed format or delimited ASCII file, the output file specification will be written to the FILE command using the same drive (Windows), directory and name as the input file and appending the file extension *.dat*. You can tell Stat/Transfer not to use this path, but instead to use only the default directory '.' and the file name.   To do so, uncheck the option **Write Complete Paths** in the **ASCII/Text File Write** section of the **Options** dialog box or use the SET command CODE-OMIT-PATHS Y with the command processor.

FORMAT Command
The format command is optional for fixed format data, but it must be present in order to read delimited data. It begins in the first column.

When reading fixed format data, the FORMAT command, if present, is followed by 'fixed'. This command is never required for fixed format files, but you may wish to use it for documentation purposes.

When reading delimited data, the FORMAT command is followed by 'delimited', which is then followed by the delimiter. The delimiter can be 'commas', 'tabs', 'spaces' or 'semicolons', or you can specify any other delimiter character by preceding it with a '\'. For instance to use the pipe character for a delimiter, type

```
format delimited \|
```

To use commas, type

```
format delimited commas
```

FIRST LINE Command
If you need to skip lines at the beginning of your data file, use this command, which starts in the first column. After the command, enter the number of the first line to be read.

For example, to start on the third line of your file, type

```
first line 3
```

NO-OPT Command
This command over-rides optimization, it should only be used if you are absolutely certain that the widths given in your schema are wide enough to be used in the output.

VARIABLES Command
The VARIABLES command must be used. It begins in the first column.

Record Specifiers

If you have more than one record for each logical case, you must indicate the start of each new record by a slash. For example, if each case is on two records, then the VARIABLES command would have the following form:

VARIABLES
    variable specifications for record one
    /
    variable specifications for record two

where '/' signifies the start of the second record.

You must specify as many records as are present, but you need not define fields on each record.

Variable Specification for Fixed Format Data

Variable Names

For fixed format input, the required elements after the VARIABLES command are a list of variable names, followed by their column locations and, for non-numeric variables, the variable type in parentheses.

Variable names must begin with a letter or an underscore. If they have embedded spaces (not recommended) they must be enclosed by single or double quotes.

The column locations begin with column one. The variable type, when necessary, is given in parentheses after the column location.

For example, a Schema file might contain the following VARIABLES command:

Variables	
ID	1-5
Age	6-7
Name	8-20 (A)
Sex	21
Birthdate	(%d-%m-%Y)

The type is not necessary when the variable type is numeric. String variables are designated by the letter 'A', and for dates and times you should use a Stat/Transfer date input format. For dates in the format 10-MAY-1950, for example, you would use (%d-%m-%Y), the example above. Note that different formats can be used for different date and time variables.

Numeric variables can be also be specified with a starting column location, and a format giving the variable width and an optional number of implied decimal places. In this case, the ending column location is not necessary. The format is given in parentheses after the starting column location.

```
variablename  begcol    (fwidth.n)
```

If a decimal point is present in the input number, it is read and used. If a decimal place is not found in the data, the number will be divided by 10 to the nth power, where n is the number of decimal placed given in the format.

For example,

```
income  6  (F10.2)
```

will read the variable 'income' beginning in column 6, with a width of 10. If the number found in the field is 2.00 the result will be 2. If the data contain 200 the result will be two as well.

The input widths, whether given explicitly or with a format giving the variable width, may be used to specify the output width when writing program files with fixed format ASCII data. In order for these widths to be used in the output file, you should check the option **Preserve Input Widths** in the **ASCII/Text Write** section of the **Options** dialog box or use the SET command PROG-PRESERVE-WIDTHS Y in the command processor.

Variable Lists   If you have adjacent variables in a record with the same widths and of the same type, you can use a list to specify them in the VARIABLES command.

The width specified by the given column range is divided by the number of variables in the list to give the width of each variable. If the number of variables does not give an integer when divided into the width specified, then an error is returned.

For example,

```
incsat housesat 22-23
```

will describe the numeric variable 'incsat' as having a width of one, located in column 22 and the numeric variable 'housesat' as having a width of one in column 23.

Variable Range   If you have adjacent variables in a record with the same widths and types and with names of the form *variablen*, you can use a range to specify them in the VARIABLES command.

For example,

```
        scale1 to scale10   24-43
```

will expand to 10 variables named 'scale1', 'scale2' ... 'scale10', each of width 2 and located starting in column 24.

As is the case with using variable lists, if the number of variables does not give an integer when divided into the width specified, then an error is returned.

Variable Specifications for Delimited Data

For delimited input data, the required elements after the VARIABLES command are a list of variable names, followed by their variable types in parentheses.

For example, a Schema file might contain the following VARIABLES command:

Variables	
ID	(A5)
Age	(F)
Name	(A16)
Sex	(F2)
Birthdate	(%Y-%m-%d)

Variable names must begin with a letter or an underscore. If they have embedded spaces (not recommended) they must be enclosed by single or double quotes.

Variable numbers (starting with 1) are written by default in version 13 of Stat/Transfer. They are optional when reading data described by a schema, but they are useful because they allow variables to be deleted or re-arranged.

The variable type is always necessary for delimited data.

String variables are indicated by the letter 'A' followed by a width. The width specification is required.

Numeric data are indicated by the letter 'F'. A width is not required for numeric data, but may be given in order to specify the output width when writing program files with

fixed format ASCII data. You can give either a width such as 'F2" or a width and implied number of decimals, such as 'F4.2'. In order for these widths to be used in the output file, you should check the option **Preserve Input Widths** in the **ASCII/Text Write** section of the **Options** dialog box.

For dates and times, you should use a Stat/Transfer date input format, as documented in Date/Time Formats - Reading. For dates in the format 1950-MAY-10, for example, you would use (%Y-%m-%d). The formats given here override those selected in the **Date/Time Formats - Reading** section of the **Options** dialog box. Note that different formats can be used for different date and time variables.

Optional Elements for the Variables Command

Missing Value Specifications
Missing values can be coded in your input data file as blank, or as one of the extended missing values '.' and '.a-.z'. See the ASCII/Text Files - Read Options for specifying extended missing values.

However, it is probably better practice to code missing values numerically in your data and then assign these numbers to 'missing' as the data are read in. This allows the missing data to become a subject of analysis.

Up to three missing value specifications can be entered for each variable, variable list or variable range. These are entered on the same line as the variable name, variable list, or variable range, following the column numbers and type (if present). The missing value specifications are entered in square brackets and are separated by commas.

A number given by itself is an "equal-to" specification. That is, if an input value is equal to the number, it will be considered missing. In addition, missing values specifications can be entered as '<=' (less than or equal) plus a number and '>=' (greater than or equal) plus a number.

For a fixed format input file, for example, suppose the values `0', `98', and `99' for the variable 'age' in the input data are all used to indicate a missing value. Then

```
    Age  6-7    [0,98,99]
    Age  6-7    [0,>=98]
    Age  6-7    [<=1,98,99]
```

are all equivalent (assuming positive values for the variable age).

For a delimited input file, this would be specified as

```
    Age (F) [0,98,99]
```

and so on.

Variable Labels

If you only have one variable on a line (not a list or a range), and wish to use variable labels, then it is most convenient to put them on the same line as the variables.

The variable labels are enclosed in curly braces, as in

```
    Age 6-7  [0,98,99] {Age of Respondent}
```

If you are using a range or a list to define several similar variables and you wish to label them, you must use the VARIABLE LABELS command, described below.

Value Label Tag

If you wish to attach labels to the values of your variables, you must define the label set using the VALUE LABELS command, discussed below. You then attach the label to a particular variable or set of variables using a tag. The tag begins with a backslash '\'.

For example, if instead of setting values to missing, you might wish to label them instead.

```
    Age 6-7 {Age of Respondent} \agelab
```

where the value label tag '\agelab' is defined using the VALUE LABEL command as shown in the example below.

VARIABLE LABELS Command

If you are using lists or range specifications in the VARIABLES command and need to define labels for those variables, then you must use the VARIABLE LABELS command. This command is followed by as many variable name / variable label pairs as needed.

For example:

```
Variable Labels
    Housesat "Satisfaction with Housing"
    Incsat  "Satisfaction with Income"
```

will label the variable 'Housesat' as "Satisfaction with Housing" and the variable 'Incsat' as "Satisfaction with Income".

If the labels contain embedded blanks, they should be enclosed in single or double quotes.

VALUE LABELS Command

You can define sets of value labels with the VALUE LABEL command. The elements are a tag, preceded by a backslash and then value/ label pairs. For instance:

```
Value Labels
    \Agelab
      0      Inapplicable
      98      "Not Ascertained"
      99      Refused
```

If the labels contain embedded blanks, they should be enclosed in single or double quotes.

DATA Command

The input data can be read from the same file as the Schema commands, rather than read from a separate file. The DATA command, which must follow all of the rest of the Schema commands, tells Stat/Transfer to treat everything that follows as data.

When the DATA command is used, the FILE command must be omitted.

Schema File Comments

You can put comments in the Schema file with two slashes. Everything that follows on the line will be treated as a comment.

```
 // this is a comment
```

An Example Schema File for a Fixed Format File

```
File "/my data files/new/big.file"
Variables
  ID     1-5 {Arbitrary ID Number}
  Age    6-7 [0,98,99] {Respondent's Age} \agelab
  Name   8-20 (A)
  Sex    21 \sex
  incsat housesat 22-23 [0,8,9] \sat
```

```
  famsat 52 [0,8,9] {Satisfaction with Family} \sat
Variable Labels
  Incsat  "Satisfaction with Income"
  Housesat "Satisfaction with Housing"
Value Labels
  \agelab  //We only need labels for missing values
        0  "Not applicable"
       98 "Not Ascertained'
       99 Refused
  \sex
       1 Male
       2 Female
  \sat
       1 "Not Satisfied"
       2 "Somewhat Satisfied"
       3 "Very Satisfied"
       8 "Not Ascertained"
       9 Refused
```

Assuming this Schema file had the name *survey.sts*, you would read it into Stat/Transfer by selecting "ASCII - Fixed format (S/T Schema)" from the user interface, and then selecting *survey.sts* for the input file name. The Schema would then point to the data file *big.file*, in the directory */my data files/new/*.

If the data file were named *survey.dat* and placed in the same directory as the Schema file, then the FILE command would be omitted, since Stat/Transfer would find it without it being named in the FILE command.

Alternatively, you could put the data in the same file as the Schema and use the DATA command at the end of the Schema, but before your data, to tell Stat/Transfer to read your data from the same file, starting in the line following the command.

Using Schemas for Delimited Data to Exert Control over your Data

There are several common problems that can be solved by this technique. For instance you may wish to:

- Change the types of your variables (for example, from numeric to strings)
- Reorder your variables
- Change the names of your variables
- Add value or variable labels
- Control the width of your variables to meet the specifications of others
- Add user-missing values

The procedure is simple. Make sure the option in "Schema Variable Numbers" option in "ASCII - Write" is checked (that is the default). Then Select "ASCII Text Delimited (S/T Schema) as the output file type. For instance if you were to write the Stat/Transfer demo worksheet file demo.wk1, you would get the following schema:

```
FORMAT delimited commas
VARIABLES
    ID                      1       (A4)
    NAME                    2       (A12)
    BIRTHDATE               3       (%m/%d/%Y)
    INCOME                  4       (F4)
    EDUC                    5       (F2)
    SEX                     6       (F1)
    RATE                    7       (F8)
    COUNT                   8       (F7)
```

Note the variable numbers (1-8). Those make it possible to arbitrarily re-arrange or delete variables. Because these data have been optimized by default, the widths for each variable are the minimum in which they can be represented. You can change their widths by making them wider, but not narrower.   But make sure you check the "General" Options to preserve numeric and/or string widths.

You can also change the types. For instance if you wanted to make the variable ID, into a number rather than a string, you would simply change (A4) to (F4).

If you wanted to add a label, you can do it using techniques described above. For instance "SEX" could become (simultaneously changing its name):

```
        Gender              6       (F1) {Sex of Respondent} /sex
```

The order of variables are easy to change, just move the lines around with your editor (but don't change the variable numbers!). For instance if we wanted a two variable output dataset with a different order, we could have:

```
VARIABLES
    COUNT                   8       (F7)
     BIRTHDATE              3        (%m/%d/%Y)
```

dBASE Files and Compatibles

Stat/Transfer will read and write dBASE III+ and IV files, and those from compatible systems such as Clipper or Alpha Four. Obsolete dBASE II files can also be read.

Standard extension: **dbf**

Reading dBASE Files

All versions of dBASE files can have indices for key fields, which are stored in separate files. Stat/Transfer ignores these indices, and treats all files sequentially.

On input, dBASE numeric data and character variables are converted in a straight-forward manner. Logical variables are converted to numbers ('True' becomes '1', 'False' becomes '0'). Memo Fields cannot be converted and will not appear on the **Variable Selection** dialog box. Deleted records are not transferred.

Writing dBASE Files

Users should be aware that dBASE files are limited to 128 variables.

dBASE stores numeric data in fixed length character format. It is thus not very suitable for numbers which vary widely in magnitude, or which are either very large or very small.

When Stat/Transfer is transferring data from a system in which the width and number of decimal places are known, it uses that information to set the format of each field in the output dBASE files. For systems, such as SYSTAT, in which this information is not recorded in the file, Stat/Transfer sets the formats based on the target type of the variable.

Missing Data

dBASE does not directly support missing values. On input to Stat/Transfer, blanks in an dBASE file are interpreted as missing values. If a data set is being transferred to an dBASE file format, missing values in the input files are set to blank in the dBASE file. Blanks are interpreted as zero by dBASE. Many other programs, including Stat/Transfer, interpret these blanks as missing.

Output Variable Types

The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte	Number (width 4; 0 decimal places)
int	Number (width 6; 0 decimal places)
long	Number (width 11; 0 decimal places)
dBASE III float double	Number (width 16; decimal places taken from input data. If unknown, set to 4)
dBASE IV float double	Float (width 16)
string	Character
date	Date
time date/time	Character (written using ASCII format options currently in effect)

DDI (Data Documentation Initiative) Schemas

The Data Documentation Initiative (DDI) is an open, international effort to provide a standard way of describing data from the social, behavioral and economic sciences. DDI Schemas are in XML and can describe metadata across the life-cycle from questionnaire design through analysis. See www.dddalliance.org.

Stat/Transfer supports DDI 3.1 Standard. The data are contained in two files: the dictionary information is stored in a file with the extension *.xml* and the data are stored in a separate delimited ASCII file with the extension *.dat*.

Standard extension: **xml**, **dat**

Reading DDI

Stat/Transfer will read and use DDI variable and value labels. Delimited data files are supported for reading and writing. The missing values and the delimiter are taken from the Schema.

Writing DDI

Stat/Transfer writes delimited DDI. The specification requires that elements within the Schema be identified by an "agency". This is typically a url. By default it is "example.org", but you should change it to something more appropriate at **Output Options(1)** of the **Options** dialog box.

Missing Values

On input the missing value is taken from the Schema. On output it is a blank.

Output Variable Types

The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int	Short

float	Float
double	Double
date	Date
time	Time
date/time	DateTime
string	Text

Epi Info

Stat/Transfer will read and write files for Epi Info, a free statistical program developed by the Center for Disease Control. All versions through Version Six are supported.

Standard extension: **rec**

Reading Epi Info Files
The Epi Info *.rec* file contains both a dictionary with enough information for the program to construct a data-entry screen for the file, and the data itself in ASCII format. Stat/Transfer will use the data-entry description field as the variable label, if it is present.

Writing Epi Info Files
Because Epi Info files are basically formatted ASCII, they are not suitable for numbers which vary widely in magnitude. Keeping in mind that the *.rec* file is also a data-entry template, Stat/Transfer will make its best effort to make an attractive and functional one. Labels will be used, if available for variable descriptions.

Missing Data
Blank fields are used by Epi Info for missing values. Stat/Transfer recognizes this when reading Epi Info files. Missing values will be written as blanks on output to Epi Info format.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte	Number (width 4; 0 decimal places)
int	Number (width 6; 0 decimal places)
long	Number (width 11; 0 decimal places)
float double	Number (width 16; decimal places taken from input data. If unknown, set to 4)
string	Text
date	Date
time date/time	Text (written using ASCII format options currently in effect)

EViews

EViews is a popular package for econometric analysis, forecasting and simulation. Series and Series Alpha objects are supported.

Reading EViews

Stat/Transfer can read data from versions later than version 6.

All field types are supported on input: single and double precision numeric series, compressed numeric series, alpha compressed series and date, time, and date-time.

EViews dated, regular frequency workfiles are transferred to an output file in which each series will be represented by separate variables. In addition Stat/Transfer will add two new variables, "date_start" and "date-end" to reflect the frequency that is used in the file. Other workfile structures are simply transferred to tables with each series presented by a separate variable. All regular date frequencies are supported, with two exceptions, "Daily-custom week" and "intraday."
Number of the records in the files is defined by the number of observations stored in the "Range" EViews object.

Writing Eviews

Stat/Transfer writes Version 5 EViews files, which can be read by Eviews 5 or higher.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float double	Number
string	Text
date time date/time	Date/time

Excel Worksheets

Stat/Transfer will read and write files from Excel. It will read all versions and will write Version 2.1 files and files for Excel 97 and higher versions. Note that Excel 97 and higher versions support long strings (up to 32K) and more (up to 64K) records in a worksheet.

Excel 2007 has vastly increased limits over earlier versions. The maximum number of columns has been increased from 256 to 65,536 and the maximum number of rows has increased from 16,384 to over one million. Note however, this does not mean that Excel is an appropriate tool for very large datasets. You will find that the time to load very large files is agonizingly slow.

Excel 2013 has added a new "Strict Open XML" format.   Stat/Transfer also will read this format.

***Standard extension:* xls**

Reading Excel Worksheet Files
Because worksheet files are in general not designed to hold statistical data, only worksheets in certain formats can be read.

Worksheets must be in worksheet database format or in certain modifications of this. It is most straightforward to read worksheets in database format.

Database Format
Worksheet database files are structured worksheets where each row is a single case and each column contains a variable. Data can consist of numbers (including serial date numbers), labels, or formulas.

The first non-blank row of a worksheet database file usually has strings in each column that give the names of the variables. The data then begins in the next row. However, variable names may be in different rows or not present at all.

You have several options to specify what part of an input worksheet to read and how to read variable names. An option allows you to read variable labels from the row after the variable names.

Data Range
You can choose different ranges to be read in input worksheets by using the drop-down menu for **Data Range** in the **Worksheets** section of the **Options** dialog box or using the SET command, WKS-DATA-RANGE.

If you use **Autosense** (the default), Stat/Transfer will read to the first non-blank cell and use that as the upper left corner of the data range. It will then read data until it encounters an entirely blank line. This is the default behavior. You can change the behavior with respect to blank rows by using the **Blank Rows** option.

Rather than have Stat/Transfer automatically sense the number of rows to be read, you can use the other options for **Data Range** to specify a range, either by giving a named range or by giving explicit coordinates.

When a range has been specified, Stat/Transfer's treatment of entirely blank rows will also be overridden. They will be returned in your output data and, in addition, blank rows at the end of the worksheet, through the last row of the specified range, will also be returned.

Determining Variable Names
By default, Stat/Transfer will attempt to determine whether or not the data in the first non-blank row (or the first row of a specified range) are variable names or the first row of data. It does so by looking for at least one column in which there is a string in the first row and a number in the second. If this behavior is inappropriate for your worksheet (for example, if you have only string data), you can specify different options in the **Field Name Row** drop-down menu of the **Worksheets** section of the **Options** dialog box. You can specify that variable names must be taken from the first non-blank row, that they be taken from a specific row or that the worksheet does not contain variable names, so that Stat/Transfer should assign them ('col1' through 'col*n*').

Determining the Data Types and Widths
After identifying the label row, Stat/Transfer will look at the entire column. If any cell contains a string, the entire column will be treated as string data (with numbers and dates converted to strings). If the column is all numeric or blank, it will be treated as a numeric column.

The width of the column for each numeric variable and the format of the first non-blank data cell in that column are used, where possible, to set the default target, or output, types for the numeric variables. If the format of the first data row has any decimal places (for example, F(2)), the target type will be 'float'. On the other hand, if the cell format has no decimal places (for example, F(0)) the target types will be various flavors of integers which depend on the column width. If the column width is less than three, the target type will be 'byte'. If the column width is less than five, the target type will by 'int'. Otherwise the target type will be 'long'. Any date format in the first data row will set the target type to 'date'.

The maximum width of character variables is determined by examining the widths of all of the strings in a column.

Stat/Transfer is lenient in typing variables from worksheets. If it is expecting a character variable and it encounters a number it will convert it to a string.

Combining Multiple Input Worksheet to a Single Output File
The option **Concatenate Worksheet Pages**, found in the **Worksheets** section of the **Options** dialog box allows you to combine worksheet pages into a single output file. This option is appropriate if your worksheet contains many sheets that are identical in structure. These can be then be combined into a single output file of any type.

For example, you may have a workbook that has 50 sheets, with one sheet for each state and the same variables on each sheet. If you check this box, Stat/Transfer will effectively combine the sheets into one large input file, dropping the field names, if necessary, on the second and higher sheets. You will end up with the data from all of your worksheet pages in a single output file.

Writing Excel Worksheet Files
On output, Stat/Transfer will write variable labels in the first row of the worksheet. Data values will be placed in the second and succeeding rows.

Column widths and formats will be determined by the variable information available. Dates and character variables are straightforward. For numerical data, information on the width and number of decimal places of variables, where available, is used to set the column widths and formats.

Missing Data
On input, blank cells and cells containing labels consisting of a single dot are read as missing.

If there is a string in your worksheet, such as 'NA', that you would like to have treated as a numeric missing value, you can specify it using the **Numeric Missing Value** option in the **Worksheets** section of the **Options** dialog box.

When transferring data to worksheets from other formats, missing values will be written out to worksheets as blank cells.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int	Numeric cell (formatted if information available)

long float double	
string	Label
date	Serial date number (with date format)
time	Time fraction (with date format if available)
date/time	Date number + time fraction (with date/time format if available)

Feather

Feather is a fast on-disk format for data frames for R and Python. For more information see https://bit.ly/2EKN4ZD.

Reading Feather

Feather stores data uncompressed in column-wise order, in a dataframe that is similar to r.

Writing Feather

Stat/Transfer converts its internal type to the following Feather types

Target Type	Output Type
Byte Float Double	BYTE
Int Long	INT32
Float	FLOAT
Double	DOUBLE
Date	DATE
Time	TIME
Date/time	TIMESTAMP
String	UTF-8 String

FoxPro Files

Stat/Transfer will read and write FoxPro files.

***Standard extension*: dbf**

Reading FoxPro Files

FoxPro files can have indices for key fields, which are stored in separate files. Stat/Transfer ignores these indices, and treats all files sequentially.

On input, FoxPro numeric data and character variables are converted in a straightforward manner. Logical variables are converted to numbers ('True' becomes '1', 'False' becomes '0'). Memo Fields cannot be converted and will not appear on the **Variable Selection** dialog box. Deleted records are not transferred.

Writing Fox Pro Files
Users should be aware that FoxPro files are limited to 128 variables.

FoxPro stores numeric data in fixed length character format. It is thus not very suitable for numbers which vary widely in magnitude, or which are either very large or very small.

When Stat/Transfer is transferring data from a system in which the width and number of decimal places are known, it uses that information to set the format of each field in the output FoxPro files. For systems, such as SYSTAT, in which this information is not recorded in the file, Stat/Transfer uses sets the formats based on the target type of the variable.

Missing Data
FoxPro does not directly support missing values. On input to Stat/Transfer, blanks in a FoxPro file are interpreted as missing values. If a data set is being transferred to an FoxPro file format, missing values in the input files are set to blank in the FoxPro file. Blanks are interpreted as zero by FoxPro. Many other programs, including Stat/Transfer, interpret these blanks as missing.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte	Number (width 4; 0 decimal places)
int	Number (width 6; 0 decimal places)
long	Number (width 11; 0 decimal places)
float double	Float (width 16)
string	Character
date	Date

time date/time	Character (written using ASCII format options currently in effect)

Gauss Files

Stat/Transfer will read and write Gauss data sets. There are two Gauss formats.

1. *Gauss 89* was used on PC platforms, and consists of two files: a data file with a *.dat* extension and a header, or dictionary file with a *.dht* extension.
2. *Gauss 96* was first used on Unix platforms and now is used on all platforms. It is written to a single file with a *.dat* extension.

***Standard extension*: dat**

Reading Gauss Files

When you wish to transfer data from a Gauss data set, give Stat/Transfer the name of the data file (the file with the *.dat* extension). If Stat/Transfer can find the *.dht* file in the same directory, it will read the data file as a Gauss 89 file. If no *.dht* file is present, the data file will be read as a Gauss 96 file.

Stat/Transfer will read numeric data from integer, single precision and double precision Gauss data sets. Character data will only be read from double precision data sets.

Writing Gauss Files

On output, you can choose whether to write a Gauss 89 or Gauss 96 file. If you choose to write a Gauss 89 file, both of the Gauss files, the data file and the header file, will be written. Stat/Transfer will show the data file name, with the *.dat* extension, in the output **File Specification** line. However, the header file will be created as well, with a *.dht* extension.

Stat/Transfer writes double precision Gauss data sets.

Missing Data

Gauss supports missing values.

Output Variable Types

The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float	Double precision number

double	
string	Character (8 byte maximum)
date	MM/DD/YY format (written as a character variable)
time date/time	Character (written using ASCII format options currently in effect)

GenStat Files

Genstat is a comprehensive statistical and data management package originally conceived and developed at the Rothamsted Experimental Station in New Zealand.

Reading Genstat

Stat/Transfer will read and write both compressed and uncompressed workbooks containing multiple spreadsheets.

Writing Genstat

Stat/Transfer writes uncompressed Genstat workbooks.

Target Type	Output Type
Byte Int Long Float Double	Double
Date	Date
Time	Time
Date/time	Date/Time
String	String

gretl

gretl (Gnu Regression, Econometrics and Time-series Library) is a free, open-source package for econometric analysis.   Stat/Transfer can read and write gretl cross-section files. It does not support Time series and panel data structures.

gretl supports numeric variables only (with the exception of a single, alphanumeric, or date row label).

Reading gretl

If necessary, Stat/Transfer will decompress gretl files automatically. The character encoding is read from the file and will override any encode set in the options. If the file contains row labels, Stat/Transfer will attempt to convert them to dates using the scan format set in the Date/Time Formats Reading options.   If the row labels cannot be converted to dates, they will be treated as an ordinary string variable. The row label will be named *obs*.

Writing gretl

Stat/Transfer writes uncompressed gretl files with strings in Unicode (utf-8).   You can control which variable is used as the row label in Options/Output Options(2).   Note that because gretl has only a numeric data type, only one string variable can be written to the output file. All Date/Time variables, except for one used for the row label (optionally), will be written to the file as Excel dates (the number of days since December 30, 1899, with time as a decimal fraction of the day.

Target Type	Output Type
Table Byte Int Long Float Double	Number
Date	Excel date number
Time	Decimal Fraction (noon = .5)
Date/time	Date number + time
String	Not exported except for row label

HTML Tables

Stat/Transfer will write HTML tables for use in web pages.

***Standard extension*: htm**

Writing HTML Tables
Field names will be written in bold characters as the first row of the table. Values of each field are then written down the corresponding column.

The table generated by Stat/Transfer is bracketed by <HTML> tags. Thus it can be loaded directly into your browser. However, most users will probably want to cut and paste the table into a larger HTML page. To make this process easier, the output HTML tables have reasonably short lines and continuation lines are indented.

Note that HTML output is in general only appropriate for fairly small tables. Stat/Transfer will transfer large data sets to HTML format. However, if you use a large table in a web page, many browsers will be brought to their knees when they try to read the table.

Missing Data
Missing values for any data type are written as a non-breaking space, ' '. On any browser, this will cause a blank table cell to be displayed.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float double	Number (with a precision of up to 15 decimal places)
string	Character
date time date/time	Character (written using ASCII format options currently in effect)

JMP Files

JMP is a general statistics package from the SAS Institute that runs on both the Windows and Macintosh platforms. Stat/Transfer reads and writes files that are usable on either platform.

All versions higher than 3 are supported.

*Standard extension***:jmp**

Reading JMP Files
JMP allows (and encourages) variable names of up to 31 characters. These variable names can contain embedded blanks and any other character. They are thus highly likely to be truncated or altered when they are transferred to other statistical packages. Because of this, Stat/Transfer will use the JMP variable name for a variable label if it is greater than eight characters and contains an embedded blank, or if you have checked the option **Write new, numeric variable names** in the **General Options** section of the **Options** dialog box.

By default, the "notes" property for a variable will be used as a variable label if present. If you have additional textual data in the "custom properties" of your variables, you can append this to the variable labels by checking the option **Use custom properties as variable labels** in the **JMP Options** section of the **Options** dialog box.

Reading JMP files with the Command Processor
It is necessary to use a file-type tag to distinguish different versions of JMP files when reading them with the command processor.

Version 4+ files are read by default. That is, if the input is a Version 4+ JMP file that has the standard extension *.jmp,* no file-type tag is needed. If you want to read a Version 3 JMP file, then the file name must be preceded by the file-type tag *jmp3.*

Writing JMP Files
Variable labels, when available from the input file, will be written to JMP output files as variable "notes."

You can choose to have Stat/Transfer write value labels to output JMP files by checking the option **Write value labels to JMP** files in the **JMP Options** section of the **Options** dialog box.

Reading JMP files with the Command Processor

Since the same standard extension is used for all versions of JMP files, it is necessary to use a file-type tag to distinguish them when writing JMP files with the command processor.

Version 6 is written by default. That is, if the output is a Version 6 JMP file that has the standard extension *.jmp,* no file-type tag is needed.   If you want to write a Version 3 JMP file, then the file name must be preceded by the file-type tag *jmp3.*   For Version 4 and 6 JMP files, the file name must be preceded by the file-type tag *jmp4.*

Missing Values
JMP has a missing value for each numeric type. Stat/Transfer recognizes these on reading JMP files and will write them when writing JMP output.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
float double	Numeric
byte	Integer1
int	Integer2
long	Integer4
string	Character
date	Date
time	Time
date/time	Date/time

JSON-Stat

JSON-Stat provides simple, lightweight representation of tabular data. It is used by many national statistical agencies for data dissemination. See www.json-stat.org for further information. Stat/Transfer provides support for reading only.

Standard extension: *json*

Reading JSON-Stat

JSON-Stat files allow numerous objects and datasets to be stored in a single file. Stat/Transfer will return the last object in a file, provided it is a dataset. This will work for many, but not all JSON-Stat files.

Missing Values
Blank elements become missing.

LIMDEP Files

Stat/Transfer supports all versions of LIMDEP . LIMDEP is an econometric software program for the analysis of limited dependent variables.

Standard extension: **lpj**

Reading LIMDEP Files
LIMDEP has only a single data type, consisting of double precision numbers.

Writing LIMDEP Files
Stat/Transfer enforces LIMDEP's 200 variable limit.

Character variables cannot be exported to LIMDEP, since the program does not support them.

Missing Values
LIMDEP recognizes missing values.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
string	*not exported*
byte int long float double date time date/time	Number (with a precision of up to 15 decimal places)

Matlab Files

Matlab matrices through Version 5 are supported on the following platforms:
    Macintosh
    Windows & OS/2
    Unix HP/Sun/IBM

Matlab datasets for version 7+ are also supported.

Standard extension: mat

Reading Matlab Files
Stat/Transfer will automatically recognize an input file's platform of origin on input.

Matlab does not write variable names into the file. Therefore Stat/Transfer makes up a variable name for each column, 'coln', which consists of the string 'col' plus the number of the column.

All values in a given matrix are of a single type, usually double precision, although particularly large matrices may be written out by Matlab in integer format, when this is possible.

Strings are not supported in Matlab files.

Stat/Transfer does not read complex matrices.

Writing Matlab Files
On output, Stat/Transfer writes matrices that can be read by any version of Matlab. Datasets can be read on any version higher than seven.

Stat/Transfer always writes Matlab files in double precision.

Stat/Transfer does not write complex matrices.

Missing Data
Matlab supports missing values.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type

byte int long float double	Double
date	Date (written as serial day number
time	Time (written as fraction of day)
date/time	Date/time (integer gives serial day number, fractional part gives time)
string	Not written for matrices, written for data-sets

Mineset Files

Mineset is a data visualization package from Silicon Graphics, Inc. The published and portable data format is tab-delimited ASCII, stored in a file with a *.data* extension and described by a dictionary in a file with a *.schema* extension.

Standard extensions: **schema, data**
Reading Mineset files
When using Mineset data as input for a transfer, give the file with the *.schema* extension (the dictionary file) as the input file. Stat/Transfer will then look in the same directory for a file of the same name, but with the *.data* extension and will read the input data from this file.

Some of the more exotic features of Mineset files such as arrays and enumerations are not supported.

Writing Mineset files.
In the **Output File Specification** of the **Transfer** dialog box, specify a name for the *.schema* file. The data file will then be written to the same directory with a *.data* extension

Missing Values
Mineset uses '?' for missing numeric values. Stat/Transfer recognizes this on input of Mineset files and writes it on output to Mineset files.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long	Integer
float	Float
double	Double
string	String
time	Time
date	Date

date/time	

Minitab Worksheets

Minitab is a general statistics package from Minitab, Inc. Stat/Transfer will read Minitab worksheets written by Versions 8 - 14 of Minitab and will write Version 11 worksheets.

Standard extension: **mtw**

Reading Minitab Worksheets

Stat/Transfer reads Minitab columns. It does not read the constants or matrices that may be stored in Minitab worksheets.

Version 12 of Minitab introduced a new project file type that may contain several worksheets along with other data. Stat/Transfer will read these worksheets.

Stat/Transfer reads Minitab Versions from version 8.

Writing Minitab Worksheets

Stat/Transfer writes Minitab columns to Version 11 format or in the Version 12+ worksheet format.

Missing Values
Minitab recognizes missing values.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float double	Number
string	Text
date time date/time	Date/time

Mplus Files

Stat/Transfer supports exporting to Mplus Version 6-7 for Windows. Mplus is a latent variable modeling program. See www.statmodel.com.

***Standard extension*: inp, dat**

Writing Mplus Files
Mplus datafiles consist of a dictionary with an extension *.inp*, and a delimited data file with a *.dat* extension. Character variables cannot be exported to Mplus, since the program does not support them.

Value and variable labels are fully supported.

Missing Values
Mplus recognizes missing values, which are written as a '.'.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
string	*not exported*
byte int long float double date time date/time	Number (with a precision of up to 15 decimal places). Dates are written as serial day numbers with a base of December 31, 1899. Times are written as fractions of a day.

NLOGIT Files

Stat/Transfer supports all versions of NLOGIT for Windows. NLOGIT is an econometric software program for the analysis of discrete choice data.

***Standard extension*: lpj**

Reading NLOGIT Files
NLOGIT has only a single data type, consisting of double precision numbers.

Writing NLOGIT Files
Stat/Transfer enforces NLOGIT's 200 variable limit.

Character variables cannot be exported to NLOGIT, since the program does not support them.

Missing Values
NLOGIT recognizes missing values.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
string	*not exported*
byte int long float double date time date/time	Number

ODBC Data Sources

Stat/Transfer can read or write data for any file type for which you have, or can obtain, an ODBC driver. These include such relational databases as Oracle, Sybase, DB2, or Informix, or flat file databases such as Btrieve.

Standard extension: **[none]**

Reading and Writing ODBC Data Sources
Depending on the driver, you may need to configure your data source from the Windows control panel.

When connecting to an ODBC data source, you may need to supply a password or other information. If a particular data source requires such information, you will be prompted automatically.

Note that for client server databases in particular, some configuration and installation of database-specific components may well be required. If you have difficulty connecting to a data source with Stat/Transfer, please try to connect with another tool such as Excel, or one supplied by your database vendor. If you have difficulty with another tool, you should first talk to a local database administrator or the database vendor's support organization before contacting Circle Systems for support.

Reading
For ODBC, the Stat/Transfer user interface will present a list of installed data sources, instead of the **Open** or **Save** dialog boxes. For details on using ODBC data sources with the command processor, see The DBR and DBW Commands and Running Batch Jobs with ODBC.

Stat/Transfer can either read single tables or multiple tables that are joined in a "view".

Writing
On output, tables can be created in a new file, new tables can be created in an existing file, or existing tables can be overwritten with a new table.

In addition, new data can be now appended to an existing database table. This option is off by default and must be turned on by using the **Append to Access and ODBC Tables** option in the **ODBC/Access Options** section of the **Options** dialog box or the SET command DB-TABLE-APPEND in the command processor.

Stat/Transfer will match as many variables as possible to those already in the table and add your data to the matching columns. Obviously at least one column must match

exactly and, in addition, the table must be free of constraints, such as those requiring unique keys, that would prohibit a simple "append' operation.

Missing Values
Support for missing values depends on the ODBC driver you are using. However, in most cases, missing values are supported.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int	Smallint
long	Integer
float	Real
double	Double
date	Date
time	Time
date/time	Timestamp
string	Character

OpenDocument Spreadsheets

The OpenDocument format for spreadsheets is an ISO standard XML format for spreadsheet data. It supported by numerous applications, including OpenOffice.org, LibreOffice, and Google Docs.

Standard extension: ods

Reading OpenDocument Spreadsheets
Because worksheet files are in general not designed to hold statistical data, only worksheets in certain formats can be read.

Worksheets must be in worksheet database format or in certain modifications of this. It is most straightforward to read worksheets in database format.

Database Format
Worksheet database files are structured worksheets where each row is a single case and each column contains a variable. Data can consist of numbers (including serial date numbers), labels, or formulas.

The first non-blank row of a worksheet database file usually has strings in each column that give the names of the variables. The data then begins in the next row. However, variable names may be in different rows or not present at all.

You have several options to specify what part of an input worksheet to read and how to read variable names. An option allows you to read variable labels from the row after the variable names.

Data Range
You can choose different ranges to be read in input worksheets by using the drop-down menu for **Data Range** in the **Worksheets** section of the **Options** dialog box or using the SET command, WKS-DATA-RANGE.

If you use **Autosense** (the default), Stat/Transfer will read to the first non-blank cell and use that as the upper left corner of the data range. It will then read data until it encounters an entirely blank line. This is the default behavior. You can change the behavior with respect to blank rows by using the **Blank Rows** option.

Rather than have Stat/Transfer automatically sense the number of rows to be read, you can use the other options for **Data Range** to specify a range, either by giving a named range or by giving explicit coordinates.

When a range has been specified, Stat/Transfer's treatment of entirely blank rows will also be overridden. They will be returned in your output data and, in addition, blank rows at the end of the worksheet, through the last row of the specified range, will also be returned.

Determining Variable Names
By default, Stat/Transfer will attempt to determine whether or not the data in the first non-blank row (or the first row of a specified range) are variable names or the first row of data. It does so by looking for at least one column in which there is a string in the first row and a number in the second. If this behavior is inappropriate for your worksheet (for example, if you have only string data), you can specify different options in the **Field Name Row** drop-down menu of the **Worksheets** section of the **Options** dialog box. You can specify that variable names must be taken from the first non-blank row, that they be taken from a specific row or that the worksheet does not contain variable names, so that Stat/Transfer should assign them ('col1' through 'col*n*'.)

Determining the Data Types and Widths
After identifying the label row, Stat/Transfer will look at the entire column. If any cell contains a string, the entire column will be treated as string data (with numbers and dates converted to strings). If the column is all numeric or blank, it will be treated as a numeric column.

The width of the column for each numeric variable and the format of the first non-blank data cell in that column are used, where possible, to set the default target, or output, types for the numeric variables. If the format of the first data row has any decimal places (for example, F(2)), the target type will be 'float'. On the other hand, if the cell format has no decimal places (for example, F(0)) the target types will be various flavors of integers which depend on the column width. If the column width is less than three, the target type will be 'byte'. If the column width is less than five, the target type will by 'int'. Otherwise the target type will be 'long'. Any date format in the first data row will set the target type to 'date'.

The maximum width of character variables is determined by examining the widths of all of the strings in a column.

Stat/Transfer is lenient in typing variables from worksheets. If it is expecting a character variable and it encounters a number it will convert it to a string.

Combining Multiple Input Worksheet to a Single Output File
The option **Concatenate Worksheet Pages**, found in the **Worksheets** section of the **Options** dialog box allows you to combine worksheet pages into a single output file. This option is appropriate if your worksheet contains many sheets that are identical in structure. These can be then be combined into a single output file of any type.

For example, you may have a workbook that has 50 sheets, with one sheet for each state and the same variables on each sheet. If you check this box, Stat/Transfer will effectively combine the sheets into one large input file, dropping the field names, if necessary, on the second and higher sheets. You will end up with the data from all of your worksheet pages in a single output file.

Writing OpenDocument Spreadsheets
On output, Stat/Transfer will write variable labels in the first row of the worksheet. Data values will be placed in the second and succeeding rows.

Column widths and formats will be determined by the variable information available. Dates and character variables are straightforward. For numerical data, information on the width and number of decimal places of variables, where available, is used to set the column widths and formats.

Missing Data
On input, blank cells and cells containing labels consisting of a single dot are read as missing.

If there is a string in your worksheet, such as 'NA', that you would like to have treated as a numeric missing value, you can specify it using the **Numeric Missing Value** option in the **Worksheets** section of the **Options** dialog box.

When transferring data to worksheets from other formats, missing values will be written out to worksheets as blank cells.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float double	Numeric cell (formatted if information available)
string	Label

date	Formatted Date
time	Formatted Time
date/time	Formatted Time Stamp

OSIRIS Files

OSIRIS is a general purpose statistical package written for use on IBM mainframes. It is no longer actively supported. However an enormous store of survey data are available in OSIRIS format from the Inter-University Consortium for Political and Social Research (ICPSR) at the University of Michigan. For this reason, Stat/Transfer will read, but not write OSIRIS data.

An OSIRIS data set consists of a dictionary file and a data file.

Standard extensions: **dict, data**

Reading OSIRIS data

When using OSIRIS data as input for a transfer, you should make sure that the dictionary file and the data file have the same name and the extensions *.dict* and *.data* respectively. They should also be in the same directory.

When naming the input file in the **Transfer** dialog box, you must give the file with the *.dict* extension (the dictionary file) as the input file. Stat/Transfer will then look in the same directory for a file of the same name, but with the *.data* extension and will read the input data from this file.

Stat/Transfer supports OSIRIS Type 1 data sets, which is the variant in which data are usually distributed. The dictionary file is a binary file in which strings are represented in EBCDIC notation. The data files were originally in that format, but the ICPSR generally distributes them in ASCII format. Stat/Transfer will support either of these formats and automatically senses which format your data are in.

If you are downloading an OSIRIS dictionary for use with Stat/Transfer, you should use a binary file transfer mode.

OSIRIS has variable and value labels, which are supported by Stat/Transfer.

Missing Values

An OSIRIS dictionary allows for two numeric missing values. By default, these are recognized by Stat/Transfer. However, the default behavior can changed with the **User Missing Values** selection of the **Options** dialog box.

Paradox Tables

Because Paradox stores numbers in binary rather than character representation and because it explicitly supports missing values, it is a much more suitable file format for statistical data than the dBASE format.

Stat/Transfer reads Versions 4-9 and writes a version that is compatible with 7-9.

***Standard extension*: db**

Reading Paradox Files
Paradox variable names can be up to 25 characters in length.

Paradox's date format is supported on input.

Writing Paradox Files
Stat/Transfer stores numbers into Paradox's integer format if they will fit. If not, it uses double precision representation.

Paradox's date format is supported on output.

Missing Data
Paradox supports missing values for all data types.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int	Short
long float double	Numeric
date	Date
time	Time
date/time	Timestamp
string	Alphanumeric

Quattro Pro Worksheets

Stat/Transfer will read and write Quattro Pro files.

***Standard extensions*: wq?**, **wb?**

Reading Quattro Worksheet Files
Because worksheet files are in general not designed to hold statistical data, only worksheets in certain formats can be read.

Worksheets must be in worksheet database format or in certain modifications of this. It is most straightforward to read worksheets in database format.

Database Format
Worksheet database files are structured worksheets where each row is a single case and each column contains a variable. Data can consist of numbers (including serial date numbers), labels, or formulas.

The first non-blank row of a worksheet database file usually has strings in each column that give the names of the variables. The data then begins in the next row. However, variable names may be in different rows or not present at all.

You have several options to specify what part of an input worksheet to read and how to read variable names. An option allows you to read variable labels from the row after the variable names.

Data Range
You can choose different ranges to be read in input worksheets by using the drop-down menu for **Data Range** in the **Worksheets** section of the **Options** dialog box or using the SET command, WKS-DATA-RANGE.

If you use **Autosense** (the default), Stat/Transfer will read to the first non-blank cell and use that as the upper left corner of the data range. It will then read data until it encounters an entirely blank line. This is the default behavior. You can change the behavior with respect to blank rows by using the **Blank Rows** option.

Rather than have Stat/Transfer automatically sense the number of rows to be read, you can use the other options for **Data Range** to specify a range, either by giving a named range or by giving explicit coordinates.

When a range has been specified, Stat/Transfer's treatment of entirely blank rows will also be overridden. They will be returned in your output data and, in addition, blank

rows at the end of the worksheet, through the last row of the specified range, will also be returned.

Determining Variable Names

By default, Stat/Transfer will attempt to determine whether or not the data in the first non-blank row (or the first row of a specified range) are variable names or the first row of data. It does so by looking for at least one column in which there is a string in the first row and a number in the second. If this behavior is inappropriate for your worksheet (for example, if you have only string data), you can specify different options in the **Field Name Row** drop-down menu of the **Worksheets** section of the **Options** dialog box. You can specify that variable names must be taken from the first non-blank row, that they be taken from a specific row or that the worksheet does not contain variable names, so that Stat/Transfer should assign them ('col1' through 'col*n*'.)

Determining the Data Types and Widths

After identifying the label row, Stat/Transfer will look at the entire column. If any cell contains a string, the entire column will be treated as string data (with numbers and dates converted to strings). If the column is all numeric or blank, it will be treated as a numeric column.

The width of the column for each numeric variable and the format of the first non-blank data cell in that column are used, where possible, to set the default target, or output, types for the numeric variables. If the format of the first data row has any decimal places (for example, F(2)), the target type will be 'float'. On the other hand, if the cell format has no decimal places (for example, F(0)) the target types will be various flavors of integers which depend on the column width. If the column width is less than three, the target type will be 'byte'. If the column width is less than five, the target type will by 'int'. Otherwise the target type will be 'long'. Any date format in the first data row will set the target type to 'date'.

The maximum width of character variables is determined by examining the widths of all of the strings in a column.

Stat/Transfer is lenient in typing variables from worksheets. If it is expecting a character variable and it encounters a number it will convert it to a string.

Combining Multiple Input Worksheet to a Single Output File

The option **Concatenate Worksheet Pages**, found in the **Worksheets** section of the **Options** dialog box allows you to combine worksheet pages into a single output file. This option is appropriate if your worksheet contains many sheets that are identical in structure. These can be then be combined into a single output file of any type.

For example, you may have a workbook that has 50 sheets, with one sheet for each state and the same variables on each sheet. If you check this box, Stat/Transfer will

effectively combine the sheets into one large input file, dropping the field names, if necessary, on the second and higher sheets. You will end up with the data from all of your worksheet pages in a single output file.

Writing Quattro Worksheet Files
On output, Stat/Transfer will write variable labels in the first row of the worksheet. Data values will be placed in the second and succeeding rows.

Column widths and formats will be determined by the variable information available. Dates and character variables are straightforward. For numerical data, information on the width and number of decimal places of variables, where available, is used to set the column widths and formats.

Missing Data
On input, blank cells and cells containing labels consisting of a single dot are read as missing.

If there is a string in your worksheet, such as 'NA', that you would like to have treated as a numeric missing value, you can specify it using the **Numeric Missing Value** option in the **Worksheets** section of the **Options** dialog box.

When transferring data to worksheets from other formats, missing values will be written out to worksheets as blank cells.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float double	Numeric cell (formatted if information available)
string	Label
date	Serial date number (with date format)
time	Time fraction (with date format if available)
date/time	Date number + time fraction (with date/time format if available)

R

R is a free, open-source environment for statistical computing and graphics. Stat/Transfer will read and write workspace files for R versions 2 and 3.

Standard extension: **rdata**

Reading R files

The R file format is very unstructured and allows the user to write almost anything into it. Therefore Stat/Transfer imposes a few restrictions on input files. Specifically, your data file should contain at least one of the following kinds of objects.

- two dimensional matrices

- vectors

- factors

- dataframes

- tables

Stat/Transfer can read R files in either binary or ASCII format. Compressed files are supported, as is the newer Version 3 serialization format.

Writing R files

On output, Stat/Transfer writes an R dataframe. If your input data set does not have a variable named 'rownames', Stat/Transfer will create an extra variable containing the case number, stored as an integer variable and named 'rownames'.

Factors in R consist of a vector of zero-based numeric values and a vector of string labels that are mapped onto the values. You can choose to have these written to an output file as the numeric values and their value labels or you can write them as strings. This option is controlled in the R and S-Plus Options section of the **Options** dialog box. If you are going to a package such a Stata or SPSS, that supports value labels, the first option is more appropriate.

Stat/Transfer will writes R data in any of three formats. 1) The ASCII format, which is compatible across platforms, but now regarded as obsolete. 2) Version 2 compressed binary files, and 3) Version 3 compressed, binary utf-8 workspace files.

Time Values in R

R stores dates in POSIX format, which includes information on the time zone. Because of this, Stat/Transfer cannot detect the time zone. Therefore, we indicate to R that the

dates are in GMT. Unfortunately R defaults to displaying dates using the local time zone. You can work around this with the following R command:

Sys.setenv(TZ='GMT')

This will cause R will display the dates exactly as they were displayed in your original file.

Missing Data
R supports missing values. On input, missing values are converted to the internal missing value in Stat/Transfer. On output, missing values are converted to the value appropriate for each variable type.

If you are using Version One of R, check the **Write version one string missing** option in the R and S-Plus Options section of the **Options** dialog box and the appropriate value will be written for null or missing strings. If you are using a later version of R, leave this unchecked.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long	Integer
float	Real
double	Double
string	String
date	Date
time date/time	POSIX timestamp (note the time zone is not set, so the time will be in GMT)

RATS Files

RATS (Regression Analysis of Time Series) is a general-purpose econometric and time series analysis package. Stat/Transfer can read and write RATS Version 7-8 files.

Standard extensions: rat

Reading RATS Files
Unlike the rectangular file format of many of the programs supported by Stat/Transfer, RATS stores data in separate time-series of data, each of which can have differing start dates, frequencies and numbers of observations. Stat/Transfer makes these potentially very irregular data suitable for analysis in other software.

First, every series is examined to determine which has the earliest start date. The first observation in that series will determine the earliest row in the output dataset.   If all series have the same frequency, the data will be output with missing values substituted for missing values in series, if necessary, and a new date variable will be constructed to tag each row of the file. If the series differ in frequencies, the most frequent series will determine the frequency of the output and less frequent series will have their data values duplicated, as necessary, to construct an output row.

Writing RATS Files
RATS supports only numeric types. Therefore, when one transfers data to a RATS output file, all non-numeric variables (strings, dates, date/times and times) will be ignored. Stat/Transfer will create an output file with the number of time series variables equal to the number of numeric variables in the input file.

On output, by default, Stat/Transfer will look for the first date variable in the file (typically there will be only one). Using that variable, it will determine the start date for each of the other series in the file. It will also use that variable to autosense the frequency of the data. In order for this to work, the data must be in ascending sorted order and the date variable must contain dates that can be described by the one of the supported frequencies: daily, weekly, monthly, quarterly, or annual. Stat/Transfer will use the date from the first row and apply an autosensed or requested frequency.

In autosensing frequencies, Stat/Transfer is necessarily fairly rigid and will require that, for instance, annual data must start in January, monthly data begin on the first date of the month, weekly data occur at seven-day intervals and so on.

If the default options are not suitable you can change Stat/Transfer's behavior by setting the options in the RATS section on the **Options** tab.

If your data occur with a specifiable frequency, but are less than completely regular, you can turn off autosensing and specify a daily, weekly, monthly or annual frequency.

If you have more than one date variable and want to specify a specific one for use in determining the start date and, if needed, frequency, you can select a specific variable from a list. You can select a specific frequency even if you use a date variable to determine the start date.

Finally, if your file does not have a date variable in it at all, you can specify the start date and frequency. In this case, a specific frequency must be specified.

Missing Values
Missing values are supported.
Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float double	RATS double precision number
date date/time time	Not supported as series (see text)

SAS Data Files

Stat/Transfer will read and write SAS Version 6 - 9 data files for the Windows, OS/2 and Unix platforms.   Version 6.04 for PC/DOS can be read, but not written.

***Standard extensions*:**

Version 6	
PC/DOS	**ssd**
Windows & OS/2	**sd2**
Unix HP/Sun/IBM	**ssd01**
Version 7 - 9	**sas7bdat**

Reading and Writing SAS Data Files
Large files (> 2 GB) are now supported on both input and output.

The number of variables is unlimited.

Both 32 and 64 bit files are supported on input and output.

Read and write support for the maximum character variable length (32K) has been added.

Reading SAS Data Files
Stat/Transfer will read data files written by SAS Version 6.08 and above.

Compressed files (both RDC & RLE) for Version 7 and higher can be read.

SAS data files differ significantly between platforms. Stat/Transfer will read files written by SAS for all SAS platforms except for VAX VMS and IBM MVS.

Stat/Transfer will automatically recognize the type of SAS file on input.

If you are moving SAS data files between platforms, you should be sure that you use a binary file transfer method.

User formats (value labels) can be directly read from Windows SAS catalogs and CPORT files. Thus, there is no longer any need to convert these to data sets in order to read formats.

However, in all other cases, the catalog file is not read directly and value labels must therefore be transferred to the data file according to the procedure described in SAS Value Labels.

Options to control the reading of formats are set in the **Reading SAS Value Labels** section of the **Options** dialog box.

Writing SAS Data Files
On output, Stat/Transfer will write files for all SAS platforms except for VAX VMS and IBM MVS.

When you select SAS as the output type, you now have an option to control the "data representation" of the output SAS dataset. The possible data representations correspond to sixteen of the values for OUTREP= in the SAS system.

The SAS data representation that will be used for the output file will appear next to the **Output File Type** box. If this is not the data representation that you wish to write, use the **SAS Data Representation** option in the **Output Options(1)** section of the **Options** dialog box and make the appropriate selection from the drop-down list.

When writing SAS data files, you should pick an output format that is appropriate for the version of SAS that will be reading the file.

Value labels can be transferred according to the output procedure described in SAS Value Labels.

SAS date, time, and datetime formats can be controlled by options on the "SAS-Writing" tab.

Missing Data
SAS supports the missing values 'A' - 'Z', '.' and '._'.

Input
On input, when a SAS file is transferred to a Stata file or an ASCII file with extended missing values specified, the SAS input missing values will be transferred to the equivalent ones in the output file, with the exception that any missing values '._' in the input SAS file will be written out as '.' in the output

For all other output formats, all of SAS's missing values are converted to a single internal missing value in Stat/Transfer.

Output
When either an ASCII file or a Stata file with extended missing values is transferred to a SAS file, the input missing values will transfer to the equivalent SAS ones. For other file formats, on output to SAS, missing values are set to '.', the SAS standard missing value.

For input files that support user missing values (SPSS and OSIRIS), the options **User Missing Value** and **Map to extended (a-z) missing** in the User Missing Values section of the **Options** dialog box can be used to map selected user missing values to extended missing values in the SAS output file.

For all other file formats, on output to SAS missing values are set to '.', the SAS standard missing value.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte	SAS Number - width three bytes
int	SAS Number - width four bytes
long	SAS Number - width six bytes
float double	SAS Number - width eight bytes
string	SAS Character variable (maximum length of 200 bytes
date	SAS Serial Date Number
time	SAS Time (number of seconds since midnight)
date/time	SAS Datetime (in seconds

SAS Value Labels

Stat/Transfer supports the import and export of SAS value labels.

Reading SAS Value Labels
SAS, unlike other systems, stores value labels as custom "formats," in a separate "catalog" file. Stat/Transfer can read SAS user-defined formats (value labels) directly from 64 or 32 bit Windows catalog files. This is the simplest option for those who are using Windows SAS.

In addition, Stat/Transfer can read SAS formats that are stored in several other formats. These may be more convenient if your data are coming from another SAS platform, or you simply have no control over the way the data are produced and delivered.

Stat/Transfer can read formats from catalogs stored in CPORT libraries

Stat/Transfer can read formats from an exported catalog produced by SAS using the 'cntrlout=' option of PROC FORMAT. (See "Creating a File with the PROC FORMAT Statement" below.) The files produced in this way can be stored in:

- a SAS datafile in any of the data representations supported by Stat/Transfer, including Windows and all Unix platforms.

- a SAS dataset in a Windows CPORT library.

- a Transport file, which can be produced by SAS running on any platform, including those on IBM mainframes.

.

Setting the Appropriate Options with the User Interface
You can choose whether and how formats are to be read by using the SAS Writing options of the **Options** dialog box.

Do not read formats
Check this option, which is the default, if no formats will be read.

Read directly from a catalog file
Choose this option if you have a Windows SAS catalog file containing formats and wish to read them.

If this option is checked, the entry '%ipath%/catalog.sas7bcat' will automatically appear on the SAS catalog name line. This instructs Stat/Transfer to look for a file named

FORMATS.SAS7BCAT, in the same directory as your data file. You can change the path if your file is in a different location.

Read from a catalog in CPORT library
Check this option if you have your formats in a CPORT catalog.

If this option is checked, the entry '%ipath%/%iname%.stc' will automatically appear on the CPORT library name line. This instructs Stat/Transfer to look for a file with the name (with extension *.stc*) and directory of your input file, since you will most probably be reading your formats from the same file as the data. If you are reading formats and data from separate files, then you can give the name of the file on this line. You can type in a complete file specification or you can use the macros below as part of the file specification.

 %ipath% The path, including the directory, of the input file
%iname% The name, without the extension, of the input file
%iext% The extension, without the dot, of the input file

If your formats are in a member with the default name *formats*, you need not specify anything more. If not, uncheck the **Use default catalog name** box and press the **Read Library** button. You will then be able to select the member that contains your formats.

Read a SAS datafile
Choose this option if you wish to read formats from a SAS datafile produced by SAS using PROC FORMAT. When this option is checked, you must have both the SAS input file and a separate file created by SAS that contains the formats. By default, Stat/Transfer will look in the same directory as the input file for a file named *sas_fmts*.*ext*, where *.ext* is the extension of your input file. This is the name that the file will have if you have created it by the procedure below.

If you wish to have Stat/Transfer look for a file located somewhere else or with a different name, you can change it in the SAS dataset name box. You can type in a complete file specification or you can use the macros given above as part of the file specification.

Read from a dataset in a CPORT library
Choose this option if you have your formats in a dataset in a CPORT library produced by PROC FORMAT.

If this option is checked, the entry '*%ipath%/%iname%*.stc' will automatically appear on the CPORT library name line. This instructs Stat/Transfer to look for a file with the name (with extension *.stc*) and directory of your input file, since you will most probably be reading your formats from the same file as the data. If you are reading formats and data from separate files, then you can give the name of the file on this line. You can type

in a complete file specification or you can use the macros above as part of the file specification.

If your formats are in a dataset with the default name *sas_fmts*, you need not specify anything more. If not, uncheck the **Use default catalog name** box and press the **Read Library** button. You will then be able to select the dataset that contains your formats.

Read from a SAS Transport file
Choose this option if you have your formats in a dataset in a SAS Transport file produced by PROC FORMAT.

If this option is checked, the entry '%*ipath*%/%*iname*%.tpt' will automatically appear on the Transport library name line. This instructs Stat/Transfer to look for a file with the name (with extension *.tpt*) and directory of your input file, since you will most probably be reading your formats from the same file as the data. If you are reading formats and data from separate files, then you can give the name of the file on this line. You can type in a complete file specification or you can use the macros above as part of the file specification.

 If your formats are in a subfile with the default name *sas_fmts*, you need not specify anything more. If not, uncheck the **Use default member name** box and press the **Read Library** button. You will then be able to select the subfile that contains your formats.

Setting the Appropriate Options with the Command Processor Interface
SET commands are used to choose whether and how formats are to be read. See the section **SAS -Reading** in the Available Options section.

Creating a File with the PROC FORMAT Statement
To create the new file for Stat/Transfer to use for reading your SAS value labels, you will need to go into SAS and run the following small program:

```
libname mylib 'path';
proc format library = mylib
cntlout = mylib.sas_fmts;
run;
```

where 'path' is the directory that contains your input data file.

This procedure creates a SAS file in the directory 'path' that has the format information for each SAS data file. In this case, the file will have the name *sas_fmts*.*ext*, where *.ext* is the extension of the input SAS file, and it will be found in the same directory as the input file.

To put your formats in CPORT library, use PROC FORMAT after the above procedure. To create a format library in a Transport file, use the same procedure as above but reference the XPORT engine in the output 'libname' statement.

Transferring Value Labels with Data

When you carry out a transfer with a SAS data file as input, Stat/Transfer will check to see if you have checked the option **Read formats directly from a catalog file**. If so, the value labels will be transferred automatically from the catalog file named in the **Filename** box.

If you check any of the other options, Stat/Transfer will look for the file that you have specified and the formats will be transferred automatically.

Restrictions on Importing Value Labels

SAS catalog files not only support conventional value labels (the one-to-one mapping of a string to a single number), but also the mapping of a range of numeric values to a single string (for example, zip code mapped to state).

Because this latter form has no analog in any of the packages supported by Stat/Transfer, only conventional one-to-one value labels are imported from SAS.

Options controlling how labels are processed

You can control how Stat/Transfer will behave if all does not go smoothly.

Continue if the format file is not found

By default, processing will stop if a file containing the formats is not found. Change this behavior by unchecking this option.

Continue is there is an error processing formats

By default, processing will stop if there is an error reading the format file or if no tags in the dataset are matched to formats in the file. By checking this option, you tell Stat/Transfer to continue processing even if there is an error processing the formats.

Writing SAS Value Labels

SAS stores value labels in a separate catalog file that can be shared among several datasets. Using Stat/Transfer, you can choose to write value labels in two ways:

1. Direct Stat/Transfer to write a *proc format* program. This is an SAS program that will create an SAS catalog. This is more appropriate if you need to update an existing catalog

or write a catalog for a non-windows platform.
2. Direct Stat/Transfer to write a catalog directly.

Writing a Proc Format Program
1. Go to the **Options** tab.

2. In the SAS - Writing section, check **Write a Proc Format program.**

By default, the filename will have the same name as the output SAS data file, with the extention .**sas**. It will be placed in the same directory as the output SAS data file. If you want a different filename, enter it in the **Filename** field. See SAS Writing for details.

The resulting program has PROC FORMAT and MODIFY statements necessary to create the SAS catalog file. Once the program file has been created, you can run it in SAS.

Writing a Catalog File
1. Go to the **Options** tab.

2. In the **SAS - Writing** section, check **Write a Catalog File.**

By default, the filename will have the same name as the output SAS data file, with the extention .**sas7bcat**. It will be placed in the same directory as the output SAS data file. If you want a different filename, enter it in the **Filename** field. See SAS Writing for details.

Note: If your catalog is named anything other than format.sas7bcat (which is Stat/Transfer's default behavior) you will need to tell SAS how to find it with a LIBRARY statement.

SAS CPORT

The SAS CPORT is primarily used for transporting data libraries between machines. Stat/Transfer will read, but not write, Windows CPORT files for Versions higher than SAS Seven.
Standard extension: **stc**

Reading CPORT Files
CPORT files can contain multiple datasets as well as catalogs that contain value labels. When using the user interface, if you select a CPORT file as input, Stat/Transfer will display a **Member** selection line below the input **File Specification** line of the **Transfer** dialog box (see Selecting Members of SAS CPORT and Transport Files). You should select the member from which you want to read your data here.

If you are using the command processor, you select the member by using the '/Tmembername' switch with the Copy command (see Options for Pages and Tables).

If you want to read value labels from the same CPORT file, you can do so by setting the appropriate options (See SAS Value Labels).

Missing Data
Stst/Transfer supports the SAS missing values 'A' - 'Z', '.' and '._'.

On input, when a SAS file is transferred to a Stata file or an ASCII file with extended missing values specified, the SAS input missing values will be transferred to the equivalent ones in the output file, with the exception that any missing values '._' in the input SAS file will be written out as '.' in the output

For all other output formats, all of SAS's missing values are converted to a single internal missing value in Stat/Transfer.

SAS Transport Files

Stat/Transfer will read and write data sets in the SAS Transport Format. This is, according to the SAS Institute, the "best overall format" for interfacing with other systems because it is consistent across all host environments.

If you are downloading or uploading SAS data between computers, be sure to use an error-correcting file transfer protocol that is suitable for binary files.

***Standard extension*: xpt**

Working with Transport Files within SAS
The method for writing (and reading) transport files within SAS unfortunately varies across versions of SAS.

For users of release 6.x and higher, the file can be written by any DATA or PROC step that creates SAS data sets and, similarly, it can be read by any DATA or PROC step. Most commonly, PROC COPY is used to write (or to read) transport data sets. If you are using Version 6.06 or higher, you can read or write transport files by using the XPORT engine. To do so, you must name the XPORT engine in the LIBNAME statement.

For example, in Version 6.06 and higher the following code will write a transport file:

```
/*   read system file 'old' - write transport file 'trans'
*/
     libname old file-specification;
     libname trans xport file-specification;
     proc copy in=old out=trans;
     run;
```

The resulting transport file can then be used for a Stat/Transfer data transfer.

If a transport file has been produced by Stat/Transfer, it can be read in SAS with the following:

```
/*   read transport file 'trans' - write system file 'new'
*/
     libname trans xport file-specification;
     libname new file-specification;
     proc copy in=trans out=new ;
     run;
```

Note that you should ***not*** use PROC CPORT to write files that are to be read by Stat/Transfer. This procedure creates files in an entirely different and incompatible format.

Reading SAS Transport Files
More than one data set may be stored in a single transport file. If Stat/Transfer finds more than one data set in a file, it will allow you to select the one you want.

Writing SAS Transport Files
When Stat/Transfer writes a SAS transport file, it uses the file name of the input file, without the extension, as the internal name for the data set in the output file. Stat/Transfer will write only one data set to each output file.

Missing Data
SAS supports the missing values 'A' - 'Z', '.' and '._'.

Input
On input, when a SAS file is transferred to a Stata file or an ASCII file with extended missing values specified, the SAS input missing values will be transferred to the equivalent ones in the output file, with the exception that any missing values '._' in the input SAS file will be written out as '.' in the output. For all other output formats, all of SAS's missing values are converted to a single internal missing value in Stat/Transfer.

Output
When either an ASCII file or a Stata file with extended missing values is transferred to a SAS file, the input missing values will transfer to the equivalent SAS ones.

For input files that support user missing values (SPSS and OSIRIS), the options **User Missing Value** and **Map to extended (a-z) missing** in the **Options** dialog box can be used to map selected user missing values to extended missing values in the SAS output file.

For all other file formats, on output to SAS missing values are set to '.', the SAS standard missing value.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte	SAS Number - width three bytes

int	SAS Number - width four bytes
long	SAS Number - width six bytes
float double	SAS Number - width eight bytes
string	SAS Character variable (maximum length of 200 bytes
date	SAS Serial Date Number
time	SAS Time (number of seconds since midnight)
date/time	SAS Datetime (in seconds

S-PLUS Files

Stat/Transfer will read and write S-PLUS data sets. Files written on 64 bit machines such as the DEC Alpha are not supported.

***Standard extension*:** [**none**]

Reading S-PLUS files
Because the S-PLUS file format is so unstructured that it allows the user to write almost anything, including code, into it, Stat/Transfer imposes a few restrictions on input files. Specifically, your data should be in one of the following formats:

    two dimensional matrices
    vectors
    factors
    dataframes

S-PLUS writes out its data in the native format of the machine on which it is running. This means that both the byte order and the width of numbers can vary between machines. On input, Stat/Transfer will automatically sense the byte order of the machine that wrote the file.

Writing S-PLUS files
On output, Stat/Transfer writes a S-PLUS dataframe. If your input data set does not have a variable named 'rownames', Stat/Transfer will create an extra variable containing the case number, stored as an integer variable and named 'rownames'.

Factors in S-PLUS consist of a vector of zero-based numeric values and a vector of string labels that are mapped onto the values. You can choose to have these written to an output file as the numeric values and their value labels or you can write them as strings. This option is controlled in the R and S-Plus Options section of the **Options** dialog box. If you are going to a package such a Stata or SPSS, that supports value labels, the first option is more appropriate.

You can choose whether you want to write out a file with low to high byte order, appropriate for such processors as Intel or DEC, or a file with high to low byte order, for such processors as SPARC, HP, or Motorola. If you are using the Windows version of S-PLUS, select Intel (low to high) byte order on output.

Missing Data

S-PLUS supports missing values. On input, missing values are converted to the internal missing value in Stat/Transfer. On output, missing values are converted to the value appropriate for each variable type.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long	Integer
float	Real
double	Double
string	String
date	Date
time date/time	Character (written using ASCII format options currently in effect)

SPSS Data Files

Stat/Transfer will read and write SPSS data files from the following platforms:

> Macintosh
> Windows and OS/2
> Unix HP/Sun/IBM

Standard extension: **sav**

Reading and Writing SPSS Data Files
Long variable names, used in Version 12 and above, are supported.

Long strings, used in Version 13, are read and written.

SPSS 21 *gsav* compressed files are read.

Reading SPSS Data Files
Stat/Transfer automatically recognizes a file's platform of origin on input.

Stat/Transfer will read both compressed and uncompressed SPSS data files.

Writing SPSS Data Files
On output, Stat/Transfer allows you to choose a file type for Windows and OS2 or a Unix file type either for the general group of HIGH-LOW (Sun, HP, IBM and Macintosh) or LOW-HIGH byte order machines (DEC).

Stat/Transfer always writes compressed files (which on typical survey data are notably smaller).

Value and variable labels are fully supported.

Missing Data
SPSS allows for three numeric missing values. By default, on input these are recognized by Stat/Transfer. The default behavior can be changed with the User Missing Values selection of the **Options** dialog box.

By default (when the option **Map to extended (a-z) missing** is left unchecked), all user missing values that are selected according the options **Use All/Use First/Use none** will go to a single missing value which will then be converted to the "system" missing value in the target package ('.' in SAS or Stata, for example).

If the option **Map to extended (a-z) missing** is checked, user missing values will be mapped, if possible, to extended missing values in formats that support them (SAS, ASCII, or Stata).

If possible, the first letter of the value label will be used as the missing value. For instance, if the value '0' is a user missing value and is labeled as "inapplicable", it will be mapped to '.i'. This mapping will only occur for missing values that are computed with an equal operator.

If there is no label, or if the missing letter has already been used, the missing value will be mapped sequentially to '.a' - '.c'.

On output to SPSS, missing values are set to the SPSS system missing value.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float double	Number
string	Character
date	Number (formatted as an SPSS date)
time	Number (formatted as an SPSS time)
date/time	Number (formatted as an SPSS date/time)

SPSS Portable Files

SPSS Portable files (previously called Export files) were designed to transfer SPSS data sets between different kinds of computers. You can use them to move your data to and from mainframe SPSS-X and SPSS for the PC.

Standard extension: **por**

Reading and Writing SPSS Portable Files
Long variable names, used in Version 12 and above, are now supported.

Long strings, used in Version 13, are read and written.

Reading SPSS Portable Files
Mainframe Portable files should be transferred to your PC using an error-correcting communications protocol. It is quite difficult to check these files visually for errors and certain errors may fatally affect the ability of Stat/Transfer to interpret the file.

Writing SPSS Portable Files
When Stat/Transfer writes Portable files, it does so with up to ten base-thirty digits of precision.

Missing Data
SPSS allows for three numeric missing values. By default, on input these are recognized by Stat/Transfer. The default behavior can be changed with the User Missing Values selection of the **Options** dialog box.

By default (when the option **Map to extended (a-z) missing** is left unchecked), all user missing values that are selected according the options **Use All/Use First/Use none** will go to a single missing value which will then be converted to the "system" missing value in the target package ('.' in SAS or Stata, for example).

 If the option **Map to extended (a-z) missing** is checked, user missing values will be mapped, if possible, to extended missing values in formats that support them (SAS, ACSII, or Stata).

If possible, the first letter of the value label will be used as the missing value. For instance, if the value '0' is a user missing value and is labeled as "inapplicable", it will be mapped to '.i'. This mapping will only occur for missing values that are computed with an equal operator.

If there is no label, or if the missing letter has already been used, the missing value will be mapped sequentially to '.a' - '.c'.

On output to SPSS, missing values are set to the SPSS system missing value.

Output Variable Types

The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float double	Number
string	String
date	Date
time date/time	Character (written using ASCII format options currently in effect)

Stata Files

Stat/Transfer will read and write data for any version of Stata including versions running on Unix and Macintosh.

***Standard extension*: dta**

Reading Stata Files
Stat/Transfer can read data from any version of Stata. Character variables and dates are fully supported. Variable and value labels are transferred in and out of Stata.

Stata Version 13 includes a new *strl* data type. These are potentially very long, variable length strings. Stat/Transfer will read these variables. However, strings longer than the Stat/Transfer limit of 32,000 characters will be truncated and binary strL's will be skipped.

Stata Version 14 supports Unicode and the input encoding will be automatically set to utf-8.

Writing Stata Files
Versions of Stata higher than Version 7 come in two flavors (Standard or "intercooled" Stata and Stata SE or Special Edition). These differ in their limits for the number of output variables.

You can choose the "flavor" of the Stata output file from the list given in the **Output File Type** box in the **Transfer** dialog box.

You use the option **Stata Version** in the **Output Options (1)** section of the **Options** dialog box to choose the version of the Stata files written by Stat/Transfer. The default is Stata Version 16. Change this option if a different version will be used to read the file.

Any variable and value labels present in the input data set will be written to Stata files. Variable labels longer than Stata's eighty character maximum are now written to Stata both as truncated variable labels and, in full, as Stata notes.

Stata Version 13 includes a new strl data type. These are potentially very long, variable length strings. Stat/Transfer will write strings longer than a threshold value to Stata strl's rather than strings. The threshold, by default, is 32 characters, but you can change this on the Options (1) screen. Strings longer than Stata's width limit of 2045 characters will always be written to strls. Dates are written to Stata's internal date format.

The output encoding for Stata version 14 will be automatically set to utf-8 and all data and metadata such as variable names and labels will be properly transcoded.

Missing Data
Stata supports missing values.

Versions 8 and above of Stata support SAS-style extended missing values. The missing values '.A' - '.Z' and '.' are supported. The missing value '._ ' found in SAS is not used in Stata. Stat/Transfer supports these extended missing values.

When an Stata file with extended missing values is transferred to a SAS or ASCII file, the input missing values will transfer to the equivalent SAS or ASCII ones. When a SAS file or ASCII file is transferred to an Stata file with extended missing values specified, missing values will transfer to equivalent ones, except that '._' in input SAS files is written out as '.' in the output.

For input files that support user missing values (SPSS and OSIRIS), the options **User Missing Values** and **Map to extended (a-z) missing** in the **Options** dialog box can be used to map selected user missing values to extended missing values in the Stata output file.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte	Byte
int	Int
long	Long
float	Float
double	Double
date	Stata Date
time	Float (fractional part of a day)
date/time	Double (Stata date and fractional part of a day)
string	Character
strl	Character or strL

Statistica Files

Stat/Transfer supports Statistica Versions 5 and 7 - 9.

Standard extension: **sta**

Reading Statistica
Stat/Transfer will read and use Statistica variable and value labels. Each column has a single missing value, which will be applied.

Writing Statistica
Statistica does not have a string type, so character variables cannot be exported.

Missing Values
Statistica has one missing value for each variable. Stat/Transfer uses this when it reads a Statistica file. When writing a Statistica file, Stat/Transfer will use a value of -9999 for missing.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float double	Number
date	Date
time	Time
date/time	Date/time
string	*Not exported*

SYSTAT Files

Stat/Transfer writes double precision SYSTAT files. It will read either double or single precision SYSTAT files.

***Standard extension*: sys**

Reading SYSTAT Files
When Stat/Transfer reads SYSTAT data sets, it processes the variable names by 1) dropping the dollar signs on character variables and 2) removing the parentheses before and after subscripts. For example, SCALE(1) becomes SCALE1.

Writing SYSTAT Files
Any variable name in the source data set which contains a left parentheses followed by a number will be transferred into a SYSTAT subscripted variable.

Users should note that the SYSTAT error message, "You are trying to read an empty file," will occur when SYSTAT cannot find a data file. Your SYSTAT files should be in the default directory.

Missing Data
SYSTAT supports missing values.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long float double	Number
string	Character
date	Date
time date/time	Character (written using ASCII format options currently in effect)

Triple-S

Triple-S, is an open standard for the transfer of survey data and its meta-data between software packages. For information on the standard and on other packages that support Triple-S see www.triple-s.org.

Stat/Transfer supports Triple-S XML Version 2. The data are contained in two files: The dictionary information is stored in a file with the extension *.xml* and the data are stored in a separate file with extension *.asc*.

Standard extension: **xml**, **asc**

Reading Triple-S
Stat/Transfer will read and use Triple-S variable and value labels. Version 2 of the standard supports both fixed format and comma-separated data files. Stat/Transfer will read either. Triple-S supports alternative sets of text for variable and value labels. These are called "modes". For example, you can have one mode for analysis and the other for interviewing. If more than one mode is found, you will be allowed to select it by using the **Table** selector or the -*T* switch in the command processor.

The data file should be in the same directory as the dictionary file (*.xml*) and should have the
 extension *.asc*.

Writing Triple-S
Stat/Transfer writes delimited Triple-S files.

Missing Values
Triple-S simply represents missing with a blank field. Numeric fields that cannot be converted to numbers are treated as missing. On output, missing values are written as blanks.

Output Variable Types
The output variable type that results from each target variable type is given in the following table:

Target Type	Output Type
byte int long	single
float	quantity

double	
date	Date
time	Time
date/time	Character (combining date and time)
string	Character

Frequently Asked Questions

Character Encoding

Q. What are Encoding Errors?

A. Stat/Transfer stores strings internally in Unicode, which is capable of storing all of characters in all languages, plus many, many other symbols. Most older character sets are of much more limited scope. For instance, the most common encoding, ASCII, is only capable of storing a handful of symbols, letters and numbers, since it has only 127 locations for characters and control codes. Other single-byte character sets double the amount of storage and allow accented characters and other useful symbols. There are a number of such single-byte character sets, for instance one is suitable for the Cyrillic alphabet and another for modern Greek.

When Stat/Transfer reads data, it converts it to Unicode either based on the settings for character sets in the encoding options, or information written in the input file. If you file does not have information on the encoding, and is in a character set that is not the default encoding used on your computer, you must tell Stat/Transfer which encoding to use. For instance, if a Greek colleague sends you a Stata dataset, you may need to select a Greek character set in order to properly read it and translate it to a Unicode based system such as Excel. If the dataset contains non-ASCII strings and you do not set the encoding properly, you will get nonsense on output.
Because all single byte characters can be mapped to Unicode, there are seldom errors on input. However, you might encounter them if you are reading multi-byte characters such as those for Japanese.

The most common problems occur on output, when sometimes a character that was read on input has no mapping to the output character set. For instance, if you read your Greek data set and attempted to write it to SAS, using your Western European machine default, there would be many encoding errors because Greek characters in Unicode cannot be mapped to a character set such as latin1.

Some problems are more surprising because it looks as if you are dealing with ASCII, but your file has some characters that cannot be represented in the output. For instance all Microsoft applications use Unicode and characters such as the left apostrophe cannot be mapped to common non-unicode character sets. The same is true for the Euro sign, which is not present in ISO-8859-1, but is present in its more modern replacement, ISO-8859-15. If any of these characters are present, you may well get an encoding error.

Q. When don't I have to worry about this?

A. Most statistical data are numbers and ASCII letters. These are properly and unambiguously represented in all character sets. If that is what is in your data, you will not encounter encoding errors.

Also, if you are reading from or writing to a Unicode based format, the problem disappears because every character can be properly represented and there is no ambiguity. Excel and Access are in this class as well as versions of SPSS higher than 17. SAS Versions 9.1 and above store information about the encoding in the file and Stat/Transfer can use that to set the encoding. Anytime that encoding information is stored in the file, this information takes precedence over your option settings.

Q. How do I fix or work around "Encoding Errors"?

A. If the problem arises from an inappropriate character set, either for input or output, you should change the character set on the Encoding Options section of the Options dialog box. For instance, if you know your data contains Cyrillic characters, simply pick an appropriate character set.

If you think the problem may be due to sporadic characters that cannot be translated, choose the 'Substitute' option rather than 'Stop'. In that case, an underscore will be substituted for characters which cannot be converted.

Sometimes, if a transfer is necessary out of a Unicode application such as SPSS or Excel to a non-Unicode application, and the dataset contains strings in many languages, that require different encodings on output, there is simply no way to work around the problem. In that case, simply set the number of permitted substitutions to a high number and proceed with your transfer. You will then see underscores for characters that cannot be represented in your output character set.

Q. What about Stata Version 13 and earlier?

A. Stata before version 14 did not support Unicode and did support other multi-byte character sets, such as those necessary for Far Eastern Languages. If you are working with a data set in which all of the strings are in a language that can be represented by single byte characters (all European languages) just choose the appropriate output encoding. However, if your dataset contains strings in Far Eastern languages or multiple languages that use different character sets, you will simply not be able to properly represent all of the strings and will need to live with underscores in your data.

We highly recommend that Stata users upgrade to get the benefits of Unicode. Now Stat/Transfer can transfer data in any language to Stata.

General Questions

Q. How big a data set can Stat/Transfer handle?

A. There is no limit on the number of cases that can be handled.

Q. How do the date formatting functions know how to write out the proper weekday and month names for languages other than English?

A. Stat/Transfer retrieves the localized day and month names from the Windows registry. If they are correct for your locality, they will be correct in Stat/Transfer.

Q. I have a large number of files to transfer. I don't want to spend my day pointing and clicking. What should I do?

A. Use the Stat/Transfer Command Processor. It is documented in your manual (and in a separate chapter in the online help for the user interface). It will let you do some extremely powerful things such as extract all of the tables from an Access database in a single command or copy a whole directory full of Excel spreadsheets to Stata files.

Q. It's the middle of the night before a crucial deadline and my transfer won't work. What should I do?

A. It's like airplane travel. If you can't get a direct flight between Chicago and Los Angeles, try to get one that stops in Dallas. Consider what formats your destination program will read and the formats Stat/Transfer will write. Or, if you are having trouble reading a file from another program, consider any different file formats that it is capable of saving. There is usually more than one route between your source and your destination.

Q. What is the best way to pass data to my colleagues?

A. For use with general purpose software, probably delimited ASCII is the best. It is the closest thing to a lingua franca of data transport. Stat/Transfer writes delimited data in accordance with the "standard" set by Excel, and that is followed by most software packages. Worksheet files, such as Excel '97, are widely supported as well and have the advantage of storing numbers in double precision. As a general purpose transport format between statistical packages, SPSS binary .sav files and Stata files will maintain your value and variable labels and missing values. They are also platform independent.

Q. I want to save my data for use in the future, what should I do?

A. If you are saving your statistical for use in the indefinite future, the best thing to do is pick one of the **ASCII file + Program** options and/or ASCII data plus a Stat/Transfer Schema. Even if the particular program is no longer available in many years, you can be assured that some statistical package will be able to read your plain ASCII data and you will have the information that is necessary to re-construct your dataset. The worst thing to do is to store your data in a binary format. Also, pick your storage media carefully. Those who stored data on nine track tapes and decks of cards can no longer read them. We recommend ISO-standard compact disks for archival storage.

Q. I have a file in which numeric variables are stored as strings. How can I get Stat/Transfer to convert these variables to numbers in my output file?

A. Stat/Transfer will let you change strings to numbers when reading worksheets and ASCII files, but it will generally not let you do so when reading other file types. You can work around this limitation by first setting the **ASCII File Write** option **String Quote Character** to 'blank'. Then transfer your file to delimited ASCII. When you read the ASCII file and transfer it to your final destination format, your numeric variables, which were formerly stored as strings, will be numeric.

Q. I have a file in which I want some numbers to be transferred as strings. How can I do that?

A. Write it to an ASCII file with a Stat/Transfer Schema. Then edit the Schema and change the variables you want to convert to a string type. Then read the file back in.

Q. What's the best way to get support for Stat/Transfer?

A. If your question is of the form "How do I do X," please look first in the manual. If that does not solve your problem, please use our online support form. On the other hand, if you think you have found a bug in Stat/Transfer, please use the tools on the **Log** tab of the user interface. That will allow you to send us a complete description of your problem, your computing environment and the files that are necessary to reproduce the problem.

Command Processor

Q. When I go to the **Start** menu in Windows and click on **Stat/Transfer**, I see something called **Stat/Transfer Command Processor**. What is it?

A. The Stat/Transfer Command Processor is a separate program that lets you transfer files without using the user interface, but rather through simple commands. It can be invaluable if you have a large number of repetitive transfers or if you wish to do batch

transfers. It can be reached from any platform by typing 'st' at the operating system prompt.

The command processor has been integrated into the user interface, so that you can generate command files automatically and edit and run command files from the **Run Program** tab. Check the "Command Processor" section of the manual for complete details.

Q. I have an Access database with over one hundred tables. I want to convert these all to Stata files. What is the easiest way to do this?

A. First create a directory for your output files (*out*, for example). Then enter the Command Processor (see Question 1). If your input file is *c:/data/my.mdb* (assuming a Windows machine) and your new output directory was *c:/data/out*, you could use the command

```
        copy    c:/data/my.mdb    c:/data/out/*.dta    /t*
```

The '/t*' modifier in this command tells Stat/Transfer to copy all of the tables from *my.mdb* to the destination *out*. The output files will be automatically named with the name of the table and the extension *.dta*.

Q. I have fifty dBASE files I would like to move to SAS, what is the easiest way to do this?

A. Assuming your dBASE files are in */data/dbase* and you would like your output in */data/sas*, you can do this in a single command:

```
        copy    c:/data/sas/*.dbf    c:/data/sas/*.sas7bdat
```

Q. The user interface has lots of options. The command processor just has that blank prompt. How do I set some options in there?

A. You can do anything in the command processor that you can do through the user interface. The simplest way to learn to use the command processor is to run a transfer with the user interface and, when you are done, press the **Save Program** button to generate a command file for the command processor. Then examine the program from the **Run Program** tab. All of the options and commands are thoroughly documented in the online help or in the manual.

Q. How do I set my options permanently so that I don't have to enter SET commands every time I start up the command processor?

A. Put your SET commands in a file called *profile.stcmd*, located in the same directory as Stat/Transfer.

Q. I would like to run a file of options that are not the same each time. How do I run a copy command from the operating system, while at the same time executing a file of option commands.

A. Put your SET commands in any file, e.g. opt.stcmd, and then execute Stat/transfer with the special -ex switch. For example:

```
st my.xlsx out.sas7bdat -exopt
```

which will run the command in opt.stcmd before transferring my.xlsx to out.sas7bdat.

If you are a Stata user, you can use the same technique with the stcmd.ado file.

Licensing

Q. Can I install Stat/Transfer on a network server?

A. Our Single User or Workgroup licenses do not allow installation on a server. If you want to share the use of Stat/Transfer on a network, please consider a multi-user lease.

Q. Can I share the use of Stat/Transfer with my colleagues?

A. Only if the use occurs on a single machine. Please encourage the future development and support of Stat/Transfer by complying with our license agreement.

Q. How do I re-install Stat/Transfer after a disk crash?

A. It is simplest to download the latest release from our website, install it, and then re-activate it with the code you obtained from your disk envelope or original email from our sales department. If you cannot find your activation code, please visit our website at https://stattransfer.com/support/activatioin, where you can find a variety of tools to retrieve your code.

Q. How do I move my copy of Stat/Transfer from one computer to another?

A. Deactivate your software on your old computer, by pressing the appropriate button on the **About** tab. Then **s**imply re-install the software and use your activation code to activate the program. See the Activation section.
Q. Why can't I buy a "simultaneous use" license for Stat/Transfer?

A. The problem is that it takes so little time to do so much work with Stat/Transfer. If we allowed simultaneous use, one copy could cover a very large workgroup and we would not make enough to develop the next version. To figure out how many users you have in your workgroup, count the number of people who are likely to use Stat/Transfer in a one year period and buy enough licenses to cover that number.

Q. Can I use Stat/Transfer in my office and also on my laptop at home?

A. Yes, that is explicitly permitted by our license. If you are the primary user of the software you can install it on multiple machines. However more than one user cannot use the software on more than one machine. See the Activation section for details.

Linux

Q. Should I install Stat/Transfer with root privileges?

A. If you have a multi-user license and you are installing the software for others to use, it is a good idea to use the sudo command or to log in as root. In this case, Stat/Transfer will be installed in /usr/local/stattransfer16 by default. You can, of course select another location if you wish.

However, if you are installing Stat/Transfer for your own use and have a personal license (a sixteen character activation key) we recommend that you install it using your customary login id. Stat/Transfer is built so that all of its shared libraries are in known locations in subdirectories under the executable's. It thus does not require root privileges for an install and once installed, can be copied freely on your disk.

Q. I installed Stat/Transfer, but when I click on the icon, nothing happens?

A. In order to be compatible with the widest variety of environments, Stat/Transfer is built on an older Linux system and linked against relatively old libraries. Most newer distributions provide these libraries for compatibility. However some, such as Ubuntu, do not. If Stat/Transfer does not run, you can diagnose the problem by opening a terminal window, navigating to your Stat/Transfer installation directory and then typing stat-transfer. If any of the required files are missing from your system, you will then see an error message - likely one such as this:

'error while loading shared libraries: libstdc++.so.5: cannot open shared object file: No such file or directory'

If that is the case, you can install the needed library from your operating system distribution disk or from a copy found on the Internet using a tool such as Google.

ODBC Data Sources

Q. Why can't I find my database in the list of ODBC sources?

A. In order for a data source to appear in the list, you must first have installed a driver for the particular database system and, then you must have configured the driver to point to your database.

Drivers are usually supplied by your database vendor, and, particularly if you are using a client-server database, you must make sure that you have installed proper components for your particular workstation and network.

A data source, in ODBC parlance, is a driver configured to point to a particular database. For instance, you might have a driver for Oracle data and two Oracle data sources, one for each of two clinical trials you are analyzing. You can configure an ODBC driver to connect to a particular database by running the ODBC driver manager from the Windows Control Panel. Only properly configured data sources will appear in Stat/Transfers lists of available data sources.

Q. I can see my data source on Stat/Transfer's list, but I encounter errors when I try to connect to it

A. There are many possible causes for this problem. Your ODBC driver may not be properly installed or configured for your network. Your database server and/or network may be down. You might not have proper access rights or the proper password for your database.

We suggest you try to connect to your database with another tool, particularly one that is supplied by your database vendor or a general tool such as Microsoft Excel. If you still encounter difficulties, you should first seek support from your local database administrator and/or the vendor who supplied your database or driver.

Q. I am having problems with ODBC or Access on 64 bit Windows. What should I do?

A. First, you should understand the somewhat confusing way by which Microsoft implemented ODBC on 64 bit Windows. There are actually two copies of the ODBC Administration Tool. The first, which is reachable from the Control Panel, can only create and modify 64 bit data sources. The second, which is identically named, but located in an

obscure directory, creates and modifies data sources that use 32 bit drivers and which are accessible to 32 bit applications such as Stat/Transfer. You should only use the 32 bit Administration Tool, accessible from your Stat/Transfer program group, to create data sources for Stat/Transfer. Also, you must use the appropriate tool to edit existing data sources (See this Microsoft Knowledge Base article)

Second, the ODBC functions that are responsible for returning the names of data sources under 64 bit windows return both 32 and 64 bit data sources, even though 32 bit applications, such as Stat/Transfer. can only use 32 bit data sources. In order to avoid errors, Stat/Transfer only displays the data sources that it can use. Therefore, data sources that were created using the 64 bit administration tool are not visible from the Stat/Transfer menus.

Finally, if you create a data source and then cannot see it in Stat/Transfer, make sure that you have installed the 32 bit ODBC drivers.   Some applications do not routinely install these on 64 bit machines, and, to make matters worse, the 32 bit Administration tool will blindly use 64 bit drivers.

Q. How can I join two tables in a relational database?

A. Many database systems allow you to define a "view", that will appear to Stat/Transfer as a single table. If you database allows this, it is the simplest and most robust way of joining tables for Stat/Transfer. If this option is not available, the Stat/Transfer command processor allows you to submit an SQL select statement. It is simply passed through to your ODBC driver, so it must be legal SQL for your particular database driver.

SAS Data Files

Q. Will the feature that reads SAS value labels work with Unix SAS files?

A. Stat/Transfer only reads Windows catalogs, but it will read SAS formats converted to Unix datasets using Proc Format with the 'cntrlout' option. See the SAS Value Labels section.

Q. Stat/Transfer won't read my SAS data file. What's wrong?

A. First, we don't support every platform. Check the sections for SAS Data Files, SAS CPORT, and SAS Transport Files. If your SAS data are not in one of these formats, we cannot read it, and you should use SAS to create a SAS Transport file.

Further, we cannot read data that have been encrypted. If your data have been written with encryption, you must use SAS to copy the file to an unencrypted format.

Finally, if you are moving the file from another platform, make sure that you use a binary, error-correcting, file transfer protocol.

If your file is in the proper format and you still cannot read it, please check our website to see problems that have been fixed. If your problem is not there, please report the error using the mechanism on the **Log** tab of the user interface. The SAS file format has not been publicly documented and there may be aspects of it that we are not supporting properly. Please let us know about any problems you are having so that we can fix them for you and others.

Q. SAS refuses to read the SAS data file created by Stat/Transfer. What should I do?

A. First, make sure that you are writing the proper kind of file for the flavor of SAS you are using and that you are transporting the file properly. Then check our website to see if there is a problem that has already been fixed. If you think you have discovered a new problem, please let us know about it so we can fix it.

In the meantime, you can always use the Stat/Transfer output option "SAS Program + ASCII Data File" and then use the generated SAS program to read the ASCII data file into SAS.

SAS Transport Files

Q. Stat/Transfer reports a "dictionary error" or otherwise refuses to read my SAS file. What is the matter?

A. Most commonly, particularly when the file is received from others, the problem is that the file is not really a transport file, but, rather, is another kind of system file. You should examine the first part of the file, either in an editor or by simply typing it to the screen. The text "HEADER RECORD********LIBRARY HEADER RECORD" should appear at the beginning of the file. If it does not, it is not a transport file. You should refer to the Stat/Transfer manual or SAS's documentation to find out how to create a transport file in SAS.

S-PLUS Files

Q. Stat/Transfer will not read my S-PLUS file. What is wrong?

A. Remember, some S-PLUS file have very little structure and parts of the data are only meaningful to S-PLUS. Make sure that your data are in the form of a two-dimensional matrix, a list or a dataframe.

Stata Files

Q. Why do all of my labels and variable names come out in lower case when I transfer a file to Stata?

A. That's not a bug - it's a feature. We respect the "style" of the package to which we are transferring and packages such as S-Plus and Stata favor lower case letters. If you would like to maintain the case of your variable names and labels, check the option "Variable Name Case Conversions" on the "General Option" tab.

Q. Why do my string variables are come out as strL's in Stata even though they are not very long?

A. Recent versions of Stata introduced a strL type, which is useful for creating string variables longer than the 2045 character limit. In addition to allowing *much* longer strings, this variable type increases storage efficiency by only storing one instance of each string and only allocating the amount of storage that is necessary for the length of the string. Stat/Transfer, by default, will use the strL type for variables which are longer than 32 bytes. This is beneficial if you have a large number of repetitive strings. If not, it might be less efficient than a different threshold. In any case, you can set it to anything you would like (between 1 and 2045) in the Output Options (1) dialog.

Worksheet Files

Q. I have some blank rows in my worksheet. Stat/Transfer stops reading the data as soon as it encounters a blank row. Is there any way to work around this?

A. The reason Stat/Transfer behaves that way is that sometimes users like to put comments or notes at the bottom of their data block. If they put at least one blank line between the data and the comments, then by default, Stat/Transfer will read their data and skip the comments with no special actions on their part.

However if you can change the behavior in one of two ways. In the **Options** dialog box, go to **Worksheets** and you can either set the **Blank Rows** option to control reading of blank lines, or you can explicitly set a data range by using the **Data Range** option In the latter case, Stat/Transfer will return all of the rows in the range you specify. In other words, it assumes that you know what you are doing and will return blank rows if that is what you want.

Q. When I read my Excel spreadsheet, sometimes a whole column of numbers gets transferred as a string variable, even though it contains lots of numbers.

A. Stat/Transfer examines all of the cells in a column to determine the type. If there are any strings, the column will be transferred into a string variable and the numbers and dates that are in the column will be converted to their string representation. This scheme is, in general, what people expect, particularly for columns of mixed numeric and string identifiers, where the alternative strategy would make the strings into missing values.

If you have a column that you want to force to numeric, you can check it to make sure that there are not any strings or numbers formatted with a "text" format. Alternatively, you can force a type conversion by using the controls on the **Variables** dialog box.

Q. How should I represent missing values in my worksheet?

A. In general, you should just leave missing cells blank. You can also represent numeric missing data with a period. Missing data for strings can be represented by an empty string (entered with a single quote). However, blanks work just as well as any of these alternatives.

If your worksheets already use a specific string to represent missing, you can tell Stat/Transfer to use that string as a missing value by setting the Numeric Missing Value string in Worksheet options.