Center for Economic Studies

Thin Client User Guide

Integrated Research Environment (IRE) Edition

February 2018

Blank page.

Table of Contents

1 The First Day

On the first day, the RDC Administrator will help you log on to the server using a thin client and show you some basic functions. After this you should work through the first section of this guide on your own.

1.1 A Note on User ID and Passwords

Your User ID is on your Account and Data Request Form. You should memorize your User ID.

The system requires a strong password that will

- be at least 12 characters long.
- have at least 1 uppercase and 1 lowercase letter.
- have at least 1 number and 1 special character.
- NOT include your birthday, your SSN, names, any complete word (spelled forwards or backwards) or other similar elements.

Note that **your password expires if either it is older than 60 days** or **if you do not log on for 25 days**. If you will not visit the FSRDC for more than three weeks, maintain your password at https://pss.tco.census.gov – select Update my Security Questions. This will prompt you for your userid and password, thus creating a login for your account; then simply cancel the request.

1.2 Intent and Philosophy of the Thin Client User Guide

This guide is designed to familiarize you with the operating environment. The tasks set forth in Section 1 are intended (a) to make your use of the system easier and (b) to be done once and completed in one session. Although all of the tasks in this section are useful, some of them are **not optional** (denoted by the ☐ next to the task). This is a check box; once you have done the task, check it off. If you have questions, ask your Administrator.

After you have completed the tasks in this first section, we **highly** recommend that you carefully read the rest of the guide *before* you are in a pickle, frantically looking for answers. This guide is full of helpful tips and information that will make your everyday use of the system easier and if you are in a jam, it is helpful to know that you have seen the answer before and, therefore, have an idea where to find it.

The *Thin Client User Guide* is a working document: it is constantly under revision. If something is unclear, something important is omitted, or you simply can contribute something useful, please let us know about it. Send an e-mail to your administrator with “TCUG suggestion - <brief description>” as the subject.

Brief Tour of the Desktop and Customization

This section will give you the basics to navigate the thin client environment and to make changes to the default settings on the system. Since your administrator has just helped you log in for the first time, let’s start with the bar at the bottom of the screen. In Linux, this is called the panel.

1.2.1 Red Hat

The predominant icon on the panel is the Red Hat; it is the leftmost icon on the toolbar. Clicking on this icon will give you the menu for most of the applications on the system It opens to the Favorites menu. You can add apps to the Favorites menu by right clicking your most-used apps.

1.2.1.1 Set Preferences and Add Useful Buttons to the Panel

- Mouse preferences: From the Red Hat menu select System Settings → Keyboard & Mouse → Mouse → in the "General" tab select "Double-click to open files and folders" and in the "Advanced" tab change Pointer acceleration to 1.3x. Click "Apply".
- Keyboard preferences: From the Red Hat menu select System Settings→ Keyboard & Mouse → Keyboard. The menus that appear are straightforward and the most common setting is for number lock. Click "Apply" when you are done.

To make changes to the panel, right-click on it and select "Panel Options"

☐ Shell (Konsole): right-click on the panel → panel options → Add widgets → Konsole Profiles. Note that Konsole, terminal, shell, and prompt are all used as synonyms in this document.

- Lock/Logout applet: right-click on the panel → panel options → Add widgets → "Lock/Logout."

You can also delete any of the default buttons on the panel (except for the Red Hat) if you do not want them there. Simply right-click on the button you do not want and select "Delete."

1.2.2 File Manager – Dolphin

Dolphin can be opened either by clicking "File Manager" in Red Hat menu or by typing **`dolphin ~ &`** at your terminal shell's prompt.

When you open Dolphin, it will open in your home directory which starts with /home/
For important information on your home directory see section 2.3.1 on page 6.

1.2.2.1 Change and Save the View

If you do not like how the information is displayed or how much information is displayed, then from the menu click View → Adjust View Properties → View Mode: Details. Then check box to "Apply View Properties to all Folders."

1.2.3 Help Resources

There is documentation for your project-approved data set(s) in the directory for that data set in the subdirectory "Documentation." There are additional resources on the RDC intranet including a copy of the *Thin Client User Guide*.

1.2.3.1 Setup the RDC Intranet

The RDC intranet has user guides for SAS, R, Gauss and PBS Pro, as well as reference materials for Linux, the Konsole window (BASH), and much more. Here you can check server status by node and there is an increasing amount of documentation on various data sets.

The intranet URL is **http://rdcdoc.ces.census.gov**.

☐ To access the intranet, launch a web browser: Red Hat → Internet → Firefox Web Browser. Then save the RDC intranet as your homepage.

1.2.3.2 Konsole Command Information: "man"

For information on Konsole commands type **`man <command>`** [ENTER] at the prompt. (To open Konsole click on the button you created in section 1.2.1.1 on page 4.) This will bring up the **man**ual page for the specified command. You can even **`man man`** to learn about how to customize the information you receive using the **man** command. To page forward press [PAGEDOWN]; to page backward press [PAGEUP]. To exit the manual, type **`q`**.

1.3 Things that you should ALWAYS do

Following the recommendations in this section will greatly reduce the number of difficulties you and others will experience when using the RDC computer system.

1.3.1 Secure Your Session

If you walk away from your workstation, you should always lock your screen so that nobody else can access your account. Lock your screen if you will be away from your station for up to an hour. To do this, click the lock icon on the applet you created in section 1.2.1.1 or choose “Lock Session” from the Red Hat menu.

If you will be away from your workstation for more than an hour, it is important that you log out. You can do this by clicking the power button icon on the applet you created in section 1.2.1.1 or from the Red Hat menu → “Log Out…” Regardless of how you request to log off, a confirmation window will pop up. You **need to confirm** your request to log out by clicking “End Current Session.”

1.3.2 Run Debugged Programs in Batch Using PBS Pro

The most efficient use of resources is to run debugged programs as batch programs. These are submitted via PBS Pro, a job management application. For full directions on how to use PBS Pro and run batch programs, see section 3

You cannot run a process interactively overnight. Interactive sessions have a 10-hour time limit.

1.3.3 Follow Good Citizen Practices

All researchers share disk space, RAM, and processing resources on the RDC server. It is very important that everyone use only what they **need**. Good citizen practices are highlighted throughout this guide, but here, for easy reference, are a key few.

- Do program development and debugging on a subset of data.
- Interactive sessions are appropriate only for debugging programs or viewing results. (Interactive sessions time out after 10 hours)
- Once debugged, run programs in batch.
- Don't use the whole data set if you don't need to.
- Close Stata and Matlab if you are not actively using them.
- Don't open more than one interactive session of Stata.
- Save only the files you need. Compress/delete files no longer needed.

Not following "Good Citizen Practices" can result in extremely slow processing times, system problems, and poor popularity among your peers. We reserve the right to kill processes that are creating problems such as system instability or problematic resource constraints. If you have a resource intensive program, especially one that is memory intensive, please notify your Administrator in advance with the program name(s) and when you plan to run it (them) so that your Administrator can protect your process(es) from being killed and ensure smooth operation of the computing system.

2 Detailed Information on the Basics

2.1 Logging In

To get started, click the “NoMachine Enterprise Client” in the Start menu. At the next screen, double-click your project. Your username should be pre-populated. In the password field, type your password.

If the thin client has recently been rebooted, then a window will pop up that asks about a "RSA key fingerprint," click "yes."

When you log in, you will be on one of the login nodes. Analytical processes cannot be run here. PBS Pro (job scheduling software) will route your job to one of the other nodes of the cluster for processing.

2.1.1 Changing Your Password

From the start menu in VDI (outside of your NoMachine session), select Enterprise Password Self-Service or you can use https://pss.tco.census.gov from home.

2.2 Some Terminology

There are two methods to tell the computer what you want it to do: CLI (Command Line Interface) and GUI (Graphical User Interface). Most tasks can be done by either method so use whichever you prefer; however, sometimes you will need one or the other to do certain things.

Konsole, strictly speaking, is a KDE application that manages your terminal sessions; but here it is effectively synonymous with terminal, shell, command prompt, and prompt. You will use Konsole when you submit CLI instructions to the computer. This is necessary to launch programs like SAS and Stata, and to issue jobs in batch.

Dolphin is the KDE file management application, and as such is synonymous with file manager. You will use Dolphin when you want to access directories and files by GUI and to submit certain instructions to the computer.

KDE is a graphical desktop environment for Linux.

2.3 Directory Structure

Researchers have access to the following directories (a) their home directory, (b) their project directory, (c) the data directories for their approved data sets, (d) the public directories, and (e) the temporary directories. GUI access to all directories is available via Dolphin.

2.3.1 Home Directory

Your home directory is located at **/home/<a-z>/<User ID>**.

In general, you should limit the things that you save to your home directory to things that are only used by you, or things that “must” go there because that is where a program expects to find them (i.e., .bash_profile, .bashrc, autoexec.sas, sasv9.cfg, and other such files). Save **project files** (for others who need access to the files) in the “project directory.” All users’ home directories are on the same file system. If that file system becomes full, then no one can log in to the server.

2.3.2 Project Directory

To access the project directories:
CLI: From Konsole type **`cd /projects/`** [ENTER]
GUI: From Dolphin, find on the right side of the window the tab with a red folder on it, click this tab. This will bring you to the root directory. There will now be a list of subdirectories, one of which will be projects.

By default, each project directory will have 8 initial directories. The intended purpose of each of these directories are listed below. Users can create their own subdirectories inside these pre-created subdirectories. Save all project files (i.e., data sets, programs, etc.) in your project directory.

- bin – Scripts and programs for project-specific maintenance (not for use by researchers).
- data – For input/output files or information amongst project users.
- disclosure – Used for output to be reviewed by the disclosure officer.
- etc – Storing system-level configurations for per-project usage (not for researcher use).
- logs – Directory to hold logs files generated from batch jobs (and others, if desired).
- programs – Scripts and programs for the project.
- transfer – For data sets placed into the project by the data staff.
- users – Per-user directories for holding their own information, in users/{username}.

At the moment, there are no quotas on the project directories, but keep in mind that you are sharing server space with all RDC users. **Poor data management can bring the system down.** It's important to clean out data files you will not need again and to compress (using gzip) large files that will not be used in the near future.

2.3.3 Data Directories

All of the source data is located in the various classification directories (/data**/economic**, /data**/decennial**, etc.). Under the appropriate classification directory, you will find a subdirectory for each project approved survey labeled with an abbreviation for that survey. Within a survey subdirectory, you will see all the files available for the project for that dataset. You will make your own extracts from these data. Any extracts you create must be saved in your project directory. Most data directories also contain subdirectories with documentation for those data, either for the whole dataset in /data/<type>/<dataset>/doc or within specific year subdirectories for that dataset.

2.4 Managing Your Files

2.4.1 Moving, Copying, etc.

Most file management can be done with Dolphin. You can drag and drop files to move or copy them to another directory. If it is easier, then you can open **two** Dolphin windows. You can also find the commands to copy, move, and delete files and create new directories in the menus.

If you prefer to use the Konsole window, the basic file management commands are as follows. Note that you will need the prompt pointing at the appropriate directory to use FileName in the below commands, otherwise you will need to type the complete path and file name.

`man` **man** is the MOST IMPORTANT command. It is short for **man**ual and shows the help information for a given command. Syntax: **`man <command>`** [ENTER].

`cd` changes the directory you are in. Syntax: **`cd <DirectoryPath>`** [ENTER]. Simply typing **`cd`** [ENTER] will return you to your home directory and typing **`cd ..`** [ENTER] will move you up one directory.

`xpdf` opens pdf files. Syntax: **`xpdf <FileName>.pdf &`** [ENTER]

cp	copies a file. Syntax: **cp <FileName> <NewPath/FileName>** [ENTER].
ls	lists files in a directory. For more information on **ls**, see section 2.4.3.1
mkdir	Creates a new directory. Syntax: **mkdir <NewDirectoryName>** [ENTER].
mv	moves files. Syntax: **mv <FileName> <PathOfNewDirectory>** [ENTER]. You can use a wildcard to move more than one file at a time (e.g., use *.* to move all files in the current directory). For more information on wildcards, see section 2.4.3.3 on page 11.
pwd	**p**rints the full pathname of the **w**orking **d**irectory (the directory you are in).
rm	deletes specified file(s). Syntax: **rm <FileName>** [ENTER]. Wildcards are permitted as in **mv** above.
rmdir	deletes specified directory. Syntax: **rmdir <DirectoryName>** [ENTER].

Other useful commands are explained and itemized in relevant sections of this guide.

2.4.2 Permissions

Permissions determine who is allowed access to which files and directories on the system.

2.4.2.1 Determining Permissions

Each file and directory has a set of permissions assigned to it. These tell the system who has permission to see, use, and modify the files. To determine the permissions on the file, you need to list the details for each file. You can do this by setting the view in the File Manager to a detailed list (View → View Mode → Detailed List View) or by typing **ls –l** [ENTER] in a Konsole window. Using Konsole will give you a listing like this.

```
drwxr-xr-- 5 gu000999 u-la00123 512  Mar 18 00:05 dir_1
-rw-r----- 6 herna888 u-la00123 1754 Jan 23 01:29 file1
-rw------- 9 gu000999 u-la00123 5764 Jun 7  11:46 file2
-rw-rw-rw- 6 gu000999 u-la00123 496  Mar 21 09:24 file3
```

The permissions are detailed in the first column. The first space indicates the type of item (d for directory or - for a file). The next nine spaces represent sets of three types of permissions for three user types. The three types of permissions are r=read, w=write, and x=execute (they will always be listed in that order) and the three user types are owner, group, and all others.

The first set of permissions is that of the file owner. The file owner is the person with the ID listed in the third column (gu000999 and herna888). The second set of permissions is that of the group, which is indicated by the fourth column (u-la00123). The third set of permissions is that of all other possible users.

Thus, in the listing above, the first line tells us that gu000999 owns the directory called dir_1 and has full permissions (rwx). The group, u-la00123, has permission to read and execute the directory, but not to change the contents (write). Everyone outside of the group u-la00123 can read the directory. The second line shows that herna888 owns the file called file1 and has read and write permissions. The group u-la00123 has permission only to read the file. All others have no permissions. The last line shows that everyone has read and write permissions on file3.

2.4.2.2 Changing Permissions

Let's say you want someone else in your project group to be able to modify and run a file you created. You check the permissions and see that the group is assigned read-only permissions.

You will need to change the permissions to rwx for your group. Note that you can only change permissions on files or directories you own.

GUI: To change the permissions, right-click on the file and choose “Properties." Click on the tab labeled “Permissions”, and then the “Advanced Permissions” button. Here you can change the permissions severally where read=“show entries”, write=“write entries”, and execute=“enter." Then click “OK” to exit “Advanced Permissions." If you are changing permissions for a directory, you can click the box at the bottom to “Apply changes to all subfolders and their contents." Click “OK” to apply the changes and exit the “Properties” menu.

CLI: If you prefer, you may also change permissions in the Konsole window using the **chmod** command. Syntax: **`chmod (u,g,o)=(r,w,x) <FileName>`** [ENTER]

Where u=user, g=group, o=others and (r,w,x) refer to permissions read, write, and execute. If you are not already in the directory where this file is stored you must either issue a **`cd`** command to the correct directory or include the entire path name for the file.

Using the example from the previous section, let’s say gu000999 wanted to give the group read and write permissions to file2. The command for that would be **`chmod g=rw file2`**. Now gu000999 decides that people in the group shouldn’t be able to modify or execute file3 and that others should have no permissions. To set read-only permissions for the group on file3, gu000999 would type **`chmod g=r file3`**. To remove all permissions for others, gu000999 would type **`chmod o=-rw file3`**. Note the minus sign and note that two separate commands are necessary to issue dissimilar permissions to two user types. You can issue one command if permissions are the same for two or more user types.

2.4.2.3 Compressing Files

gzip is a compression command. It allows compression of individual files or the contents of entire directories easily while maintaining the properties of the file (i.e., ownership, creation date, etc.). A gzipped file will be “renamed” to include a **.gz** file extension. For example, file mystuff.log would become mystuff.log.gz once compressed using **gzip**.

CLI: To compress individual files, open a Konsole window; navigate to the appropriate directory and at the prompt type **`gzip <FileName>`** [ENTER].

To compress everything in a directory, type **`gzip –r *`** [ENTER] at the prompt. This recursively goes down through the directory tree compressing all files along the way.

gunzip is just as easy to use. Simply replace **gzip** with **gunzip** in the above instructions to undo the compression.

2.4.3 Searching for Files and Processes

2.4.3.1 Finding Files with “ls”

ls is a Linux command that lists the contents of a directory. Two common uses are (a) discover if the file the user is looking for is in a certain directory, and (b) to get an overall feel for the contents of a directory. For example, if after typing **`ls`** at the prompt, the user is sees a list of files ending with “.gif” then the user might surmise that this directory is used to store image files.

ls Syntax **`ls <options> <DirectoryPath> or <SearchTerm>`**

When **ls** is specified alone, then no options are assumed, and files are listed for the current directory. Typing **`ls`** at the prompt will simply list the contents of the current directory. Typing a

different directory path will list the contents of the specified directory. Example: **`ls /usr/bin/`** lists the contents of the directory /usr/bin.

You can also specify that only certain files are listed. Example: **`ls log`** lists a file called "log" if it exists in the current directory.

Common ls Switches
The options that accompany a command are called switches in Linux. Some of the common switches for ls are

- **`-a`** lists **a**ll of the files in a directory, including hidden files (those starting with ".").
- **`-l`** gives the list in **l**ong format, which includes information about the last date the file was modified and who has permissions to access the file.
- **`-R`** **r**ecursively lists all subdirectories in the directory.

You can also combine switches. To specify both the –a and the –l switches type **`ls -al /usr/bin/happ*`** [ENTER] at the prompt. This displays the long form of all files in the directory /usr/bin whose first four letters are happ.

Note that you cannot recursively search for a term using ls; this is why you would pipe to grep…

2.4.3.2 Piping to grep

Piping your output to grep is a clever use of commands in Linux. Here is a summary.

Piping
Piping is adding a "|" after any command. This takes the output from that command and uses it as an input for the command that follows instead of immediately displaying the output. The key combination for | is [SHIFT]+[\].

grep
grep is a command that matches information specified by the user. This is useful when you have a large amount of text that you need to search.

Piping to grep
This is how you can use these two together: Let's say you have many files in your directory and you're looking for one that you created in July. You could type **`ls -l`** [ENTER], but since you have a lot of files, that would give you several pages to look through. A more effective way to do it would be to pipe the output to grep with "Jul" as the search term. You would type **`ls -l | grep Jul`**. This way only the lines of output from **ls –l** that have Jul somewhere in the line will be displayed.

Note that to search for a file name that contains a space (we strongly discourage the use of spaces in either a file or directory name), put quotes around the search term, otherwise the space will be interpreted as indicating the start of a new input. For example, to search for a file called my file.txt, you would type **`ls "my file.txt"`**. The same applies to the grep command as well.

2.4.3.3 Wildcards

You can also use wildcards to make searches more versatile and useful.

* The star will search for any number of characters in addition to your search term.

Examples:

ls *.log will list all files that end in .log regardless of how long the file name is.
ls extract.* will list all files that start with extract regardless of the file extension. So you might see extract.log, extract.lst, extract.sas7bdat, extract.dta.gz, etc.

? The question mark will search for as many random characters as there are ?s.

Examples:
ls ????.log will list all files that end in log and have a four character file name.
ls extract.??? will list all files that start with extract and have a three character file extension. As above, extract.log and extract.lst would be listed, but extract.sas7bdat and extract.dta.gz would not.

[ ] The square brackets can be used to contain choices for a single character and lists. Important: square brackets work as a wildcard only when they are in the first position of the search term!

Example:
If you have a number of files in a directory and you would only like to list those that either start with the word “bat” or “sat” you could type **ls [bs]at***. Notice the [ ] contain the two options, in this case either “b” or “s." This search will yield results like battery.dta and satire.sas7bdat.

You can also use square brackets to contain an ordered list, for example if you have a directory full of census files called XXXXCensus where XXXX stands for the year (1997, 1892, etc.), you can list only those between 1950 and 1980 by using a list in the square brackets, ls [1950-1981]Census*.

2.4.3.4 Finding Files with “find”

The **find** command is a powerful searching tool that helps you find a file *and* its path.
find has a difficult syntax which is beyond the scope of this guide. Rather than trying to explain it completely, an example will be given along with some explanation. This should be enough to allow you to use it easily. Here's the example.

```
find –type f –name "census*" –print [ENTER]
```

This command asks the system to find *files* called census and census+something. Wildcards work within the quotes, but the remaining elements in the quotes are case sensitive. The search is started in your current directory, continuing down through the directory tree. You can't pipe your output to grep. You can match dates, file sizes, etc., and you can search for directories instead of files; however, as mentioned, the syntax is rather involved. Using the above example but replacing census* with your search term should be enough in most cases.

3 Software

We offer many applications for statistical analysis. It is important to point out, though, that we offer support for two analysis packages only: SAS and Stata. With that said, we try to provide as much information about other applications as possible. If you have familiarity with specific applications or can offer guidance that might benefit other users, then your contribution would be much appreciated.

What is available?

Application/System Component	Description
Anaconda	For Version number, please refer to the application itself, as applications are subject to updates.
Intel Composer	
Gauss	
Gurobi (& related Anaconda modules)	
Linux Operating System	
Mathematica	
Matlab	
NoMachine Terminal Server & NoMachine Enterprise Client	
OpenGeoda	
PBS Pro	
R	
Rstudio	
SAS	
Stat/Transfer	
Stata	
Stata-MP	
SUDAAN	
Tomlab, Knittro, MADD	

3.1 PBS Pro

About the RDC Server Cluster

The IRE server cluster is a group of servers that communicate with one another, but operate independently.

Only nodes scompute 1-9 and hcompute 1-12 are able to run statistical analysis applications. Nodes hpc-login1 and hpc-login2 serve as the “login nodes.” There are two types of servers or “nodes” – login nodes and compute nodes. The login nodes are the only nodes that you will have direct login access to. You will access the compute nodes via PBS Professional (PBSPro).

There are currently two login nodes: hpc-login1.rm.census.gov and hpc-login2.rm.census.gov. When setting up NoMachine Sessions to connect to the login nodes do not select them yourself. Instead use hpc-login.rm.census.gov for the server name. This will enable the system to load balance the sessions between the nodes to avoid everyone selecting hpc-login1 or hpc-login2. This scheme allows more login nodes to be added as needed.

The login nodes provide a place for you to login and set up compute jobs that you will submit to the compute nodes. You can also use the login nodes to write and edit code, check the status of jobs you’ve submitted, prepare files for disclosure review, create tables, or other “small” tasks

(things that are not computationally intensive). All of your “compute jobs” should be submitted to the compute nodes using PBSPro.

What is PBS Professional (PBS Pro)?

PBS Pro is a job scheduler that works across all nodes of the IRE server cluster. It evaluates the available resources on each node and assigns your job(s) accordingly. Use of PBS Pro will improve efficiency, stability, and maximize the resources currently available on the servers.

PBS Pro runs ALL statistical analysis jobs on the cluster – batch jobs and interactive jobs.

PBS Pro matches the resources requested by each job to the resources available on the cluster nodes, and assigns the job to a node that can accommodate the request. If there is no available node that can satisfy the request, the job is held in the queue until the resources are available to run the job. **Note – you should use batch mode except for when debugging a program.**

3.1.1 Batch Job Submission

Users run batch jobs by expressing those jobs as “job scripts” and submitting those scripts to a queue. We have pre-written scripts for some applications for ease of use. These are called “wrappers”. You can submit batch jobs by using one of the wrapper scripts for SAS, Stata and R. From the appropriate directory or using the complete pathname:

- `qsas program_name.sas &` [ENTER]
- `qstata program_name.do &` [ENTER]
- `qR program_name &` [ENTER]

Once you submit a job to PBS Pro you will see output like **`25309.hpc-app1.rm.census.gov`**
The five-digit number is the PBS Pro job identifier which is used for monitoring the process.

These wrappers may have additional parameters you can set to request more resources if your processes requires them. By default, all jobs submitted to PBS Pro are allocated 1 CPU and 5 GB of memory. To see what can be changed while still using the wrapper script, type the wrapper name on the command line by itself and press enter.

3.1.1.1 Writing Batch Scripts

You may need to write your own batch script to submit jobs to PBS Pro if you use an application that does not have a wrapper or if the wrapper cannot accommodate the parameter changes you need. Writing your own script gives you more control and allows you to use one or more “PBS directives.” (See *PBS Pro User Guide* on rdcdoc.)

All basic PBS Pro batch scripts begin with the same three lines. The second line, even though it looks like its commented out, is where you specify the PBS Pro directives for the program. In the example below we are requesting 1 CPU and 8 GB of memory and using Stata for the analysis.

The script name is example.bash and it is saved in /projects/programs:

Script text	Description

(1)	`#!/bin/bash`	(1) Specifies which kind of shell to use (bash) – never change this.
(2)	`#PBS -l select=1:ncpus=1:mem=8gb`	(2) Tells PBS Pro to expect directives and what they are. For clarity, the first item after #PBS is dash "L" and next is select equals one, ncpus is number of CPUs.
(3)	`cd /projects/programs`	(3) Change directory to full path of program file.
(4)	`stata-se -b do program_name.do`	(4) Command to launch job.

You will submit the script to PBS Pro at the command line. With the prompt pointing at the directory where example.bash is saved, type: **`qsub example.bash &`** [ENTER]

Obviously, you will change the last line of the script to use a different application. Refer to the appropriate section for information on the application you would like to use.

3.1.2 Interactive Job Submission

PBS Pro supports "interactive batch jobs." While we understand that you will need to run interactive jobs, we ask that you use batch mode whenever possible, particularly for long running jobs that you are going to leave unattended. When a batch job completes, it terminates and the resources that were allocated to it are released back into the pool where they can be used by others. An interactive batch job completes when the user types "exit" to terminate it.

To launch an interactive batch session, enter type **`qsub -I -X`** [ENTER]

This will drop you in a shell session on a compute node selected by PBS. From here, you can modify and run your programs or run GUI versions of your desired applications.

Type "exit" to exit the interactive session when you are finished. **Do not leave the interactive session running when you close NX! Also note that interactive jobs will be automatically terminated after the 12-hour time limit.**

If you need more resources than the default, you can include a resource specification. For example, **`qsub -I -X -l select=1:ncpus=1:mem=7gb`** where after **`-X`** is "dash lower case L".

3.1.3 Monitoring Jobs

When you submit a PBS Pro job, the system will return a number to identify the process. For example, **23313**.hpc-app1. You will use PBS Pro and this number to monitor your job.

To view the job queue and check the status of your job, type **`qstat`** [ENTER] at the prompt. The output will look like

```
Job id            Name    User       Time Use   S  Queue
------------      ------  ---------  ---------  -  -----
23313.hpc-app1    STDIN   macke004   00:00:00   R  interq-a
```

Column S is job status. Entries you could see in this column: **Q** means the job is waiting for resources, **R** means the job is running, **S** means the job is suspended, and **E** means exiting.

There are many options to use with qstat to get more information that is more detailed: `qstat -a` shows **a**ll jobs, `qstat -n` shows execution **n**ode, `qstat -f` shows **f**ull details, and really nice to use sometimes is `qstat -fan`.

3.1.4 Useful PBS Pro Commands

Commands

`pbsnodes -a` Gives the status of the execution nodes, shows what jobs are running on them, what resources are available, what resources are allocated, etc.

`qdel <job identifier>` Kills a running job or deletes queued job. You can only delete your own jobs.

Tips

Directory paths: When you submit a PBS Pro job, it will by default "begin" in your home directory. It is often convenient to change to the directory that you are working in so that you do not have to specify complete pathnames.

Stdout & Stderr files: When you submit a PBS Pro job, it will create a output file, jobnumber.hpc-app1.rm.census.gov.OU and possibly an error file, jobnumber.hpc-app1.rm.census.gov.ER, in the /projects/logs directory. These can be useful for troubleshooting problems, otherwise they can be deleted when the job is complete.

3.2 SAS

All data sets provided to you by CES are in SAS format. Even if you don't use SAS for your work, you'll have to use SAS to convert your data to a different format. This section briefly summarizes a few points about using SAS; extensive documentation about the application is available on the RDC intranet (see section 1.2.3.1 on page 4).

3.2.1 Getting Started

To launch a SAS batch job, at the terminal prompt type **`qsas program_name.sas`** [ENTER]

To start an interactive SAS session, at the prompt of a terminal window, type **`qsub -I -X`** [ENTER] to get to a research node, then type **`sas &`** [ENTER].

3.2.2 Libraries and Members

Libraries

SAS has its own file system that consists of "Libraries" and "Members." Libraries are user-defined "pointers" that instruct SAS where to look when you refer to a specific library; you want SAS to read data from a certain directory and save the files it creates to a directory. A **libname** statement in SAS specifies the directory in which SAS should look. This is established by typing a statement in this format. **libname <libref> '<PathNameOfDirectory>';**

For example, let's say you have a directory, /projects/data/SasFiles, in which you want to save your SAS data and programs. You need to make a new library in SAS. You call it whatever you'd like (within the rules of library names), for example library1, and tell SAS that this library references the directory /projects/data/SasFiles. To assign the library, type this in the program editor.

libname library1 '/projects/data/SasFiles';

So, during this SAS session, whenever you tell SAS to read from or save to library1 it actually reads from or saves to /projects/data/SasFiles.

Note that libnames should actually be assigned in a file called autoexec.sas, not in your programs or the program editor. See section 3.2.3 below for more on this.

Members
Within Libraries are Members; members are just the files in a Library. When you want to reference a file in SAS you need to reference it by **<LibraryName>.<MemberName>**. So if you had a SAS data set called "output1" saved in your "/projects/data/SasFiles" directory, you would reference it in SAS as **library1.output1**.

Pdata Library
A SAS libname file has been created for each project. The file resides in **/projects/lib/sas/pdata.sas**. If you include this file in your SAS code, all data files accessible to the project will be available under the libname 'pdata'.

To include the file:
%include '/projects/lib/sas/pdata.sas';

Example of libname use:
proc contents data=pdata.asm1973;
run;

3.2.3 autoexec.sas

libname and **options** statements are often included in a file called autoexec.sas. This is a user created program file residing in the user's home directory. It is automatically run when SAS is invoked. Like all SAS programs, autoexec.sas can be created using any text editor.

Important: Stating default options and librefs in autoexec.sas instead of individual programs eliminates the need to change programs when data are moved. It is recommended that ONLY **options** and **libname**s be set in the autoexec file even though any legal SAS command can be included in the autoexec file; do not run **data** and/or **proc** steps from this file. Some important options are shown in this command line.

options compress=yes mlogic mprint ls = 80 ps = 60;

The second and third options deal with macro programming. **mlogic** outputs the value of each macro in the log window as it is encountered. **mprint** prints out the program code as reflected by the value of the macro in the log statement. These two options are very useful in debugging macro programs; they are not turned on by default.

Macros are probably the most efficient way of writing SAS code that repeats itself. A well-written macro program a couple of screens long can execute hundreds of **data** steps and **proc** statements. As such, we encourage the use of macros.

Caution: turning on **mprint** for a macro program that generates large amounts of program code could fill up the log window during an interactive SAS session. This will cause SAS to stop until you manually confirm that you would like to clear out the log window. For a batch job, this will not be a problem.

3.2.4 SAS Help

Reference materials are available on the RDC intranet (see section 1.2.3.1 on page 4).

Outside of the RDC, you can access SAS online documentation. A couple good links are http://support.sas.com/documentation/onlinedoc/base/index.html and http://support.sas.com/documentation/cdl/en/allprodsproc/61917/HTML/default/a003135046.htm

3.2.5 SAS Memory Error

If you are using really big data sets in your analyses, you will sometimes need to change the memory settings in SAS. Add these lines to the bottom of your .sas program (see 3.1.1.1).

```
OPTIONS FULLSTIMER;
PROC OPTIONS GROUP=MEMORY;
RUN;
PROC OPTIONS;
RUN;
```

This will print information about resources used for the program's processes in the log file. Then line (4) of the batch script should read: **`sas <file_name> -MEMSIZE 2G -REALMEMSIZE 0 -SUMSIZE 512M -SORTSIZE 512M -SYNCHIO -NOTERMINAL -NOTHREADS -VERBOSE &`**

3.2.6 A SAS Example

Below is a very short SAS program that uses some basic SAS commands. Each line has a comment that starts with /* and ends with */ that explains what the code is doing. SAS commands end with a semicolon.

%include '/projects/lib/sas/pdata.sas';
    /* This is not necessary if this statement is already in your autoexec.sas file, but it establishes pdata libref. (This is the source library of the project datasets.) */
libname to '/projects/data/SasFiles';
    /* This is not necessary if this statement is already in your autoexec.sas file, but it establishes a libref named to. (This is the destination library of the data set in this example) */
data to.data1 (keep = var3 var17 var53 var101 var3_2);
    /* Either creates a new member called data1 in the library to or replaces an existing member by the same name. The keep statement drops any variable not specified after the equal sign. That is, only the variables var3, var17, var53, var101, and var3_2 will be present in the new data set so be sure to list any variables that you create in the keep statement, too. */
  set pdata.data1992b;
    /* Takes the data that is in member data1992b in the library pdata and copies it into member data1 in library to. */
  if var101 = 2046;
    /* Drops all observations where the variable var101 is not equal to 2046. */
  var3_2 = var3 * 1000;
    /* Generates a new variable called var3_2; for every observation in the data set, it takes the value in var3, multiplies it by 1000, and stores it in var3_2. Note that if a variable called var3_2 already exists, then this statement would overwrite its contents. */
  if var17 = 0 then var17 = 496;
    /* Searches through every observation and if the value of var17 equals zero, then the value of var17 is changed to 496. */
run; /* Executes the program; if you submit your statements without a run command, then nothing will happen. */

3.2.7 Exporting SAS Data Sets

Since all of the data provided to you by CES are SAS data sets, you will need to convert the data to a different format if you need or want to use a different program.

3.2.7.1 Creating a SAS Data Extract

For either of the two exporting instructions that follow, you will first need to create a SAS data set that selects the variables that you want in the final data set. (Prior to creating the extract, you can determine which variables you want by doing a proc contents on the original SAS data set. Let's say that you want to create this extract in your project's data directory, /rdcprojects/br/br00999/data, and name it data2export.

```
%include '/projects/lib/sas/pdata.sas';
        /* This is not necessary if this statement is already in your autoexec.sas file, but it
        establishes a libref named frompdata libref. (This is the source library of the data set in
        this exampleproject datasets.) */
libname to '/projects/data';
        /* This is not necessary if this statement is already in your autoexec.sas file, but it
        establishes a libref named to. (This is the destination data set library.) */
data to.data2export (keep = var1 var4 var73 var105);
        /* List the variables from data set data2004a that you want in the new data set after keep
        = , no separator is need. */
    set pdata.data2004a;
run;
```

3.2.7.2 Exporting the Extract to Stata

Once you have created the extract to.data2export, then you can write code to create a Stata data set. Let's say that you want to save this in the same directory and that you want to name the new data set mydata.dta. Here is the SAS program you would write.

```
proc export data = to.data2export
        /* The libname statement was already issued either in your autoexec.sas file or in the
        "Creating a SAS Data Extract" step. */
    outfile = "/projects/data/mydata.dta"
    dbms = stata replace;
        /* Notice that there is no semicolon after the first and second line of this code! */
run;
```

You then use the file as you normally would in Stata.

3.2.7.3 Exporting to other Applications

As far as we know, and contribution by users would be appreciated on this, all other programs can import .csv files. Once you have created the extract to.data2export, then you can write code to create a csv file. Let's say that you want to save this in the same directory and that you want to name the new csv file mydata.csv. Here is the SAS program will write.

```
proc export data = to.data2export
        /* The libname statement was already issued either in your autoexec.sas file or in the
        "Creating a SAS Data Extract" step. */
    outfile = "/projects/data/mydata.csv"
    dbms = csv replace;
        /* Notice that there is no semicolon after the first and second line of this code! */
```

run;

You will then need to create code to import this new data set to the application you want to use for data analysis. You need to be careful as there are known problems with this method: PPNs can inadvertently be converted to a numeric format; if a PPN starts with a zero, the zero could be removed. The exact difficulties in using this method will vary depending on your set of circumstances. You should always check your data after you've converted it in order to identify any problems. Some problems are minor and may only be annoying (such as converting a ten-digit PPN with a leading zero into a nine-digit number) others may critically effect your program.

To get basic statistics on the original data set, type the following in SAS.

proc contents data = to.data2export;
run; /* Lists basic information on the data set: size, variables, etc. ***/**
proc means data = to.data2export;
run; /* Gives basic statistics on numeric variables: mean, min, max, etc. ***/**
proc print data = to.data2export (obs=5);
run; /* Prints the first 5 observations in the data set. ***/**

You will need to write equivalent code in the application you want to use for data analysis to determine if there are any potential problems. Again, any assistance by users would be appreciated on this topic.

3.2.8 SAS Data Quality

To use SAS-DQ, you need to tell SAS where the DQ file is located. Add this line to the top of your SAS program.

%DQLOAD (DQLOCALE=(ENUSA), DQSETUPLOC='
/apps/SAS/v9.4/SASHome/SASFoundation/9.4/misc/dquality/dqsetup.txt');

If your program will not run, regardless of the error message (or lack thereof), try running your program as if you were having a memory error with SAS. Instructions are specified in section 3.2.5 on page 17.

3.3 Stata

3.3.1 Getting Started

To launch a Stata batch job, at the terminal prompt type `qstata program_name.do` [ENTER]

To start an interactive Stata session, at the prompt of a terminal window, type **`qsub -I -X`** [ENTER] to get to a research node, then type **`xstata-se &`** [ENTER].

Stata-MP is available, but there are **only five** licenses. To use it, follow the instructions above by substituting mp for se. Use your good judgement as to whether you **need** to use mp or if se is suitable. If in doubt, use Stata SE.

3.3.2 Stata Frames and .do Files

Stata Frames
Stata opens as one large window that is divided into several sections termed "frames."

Output frame	Contains commands you send to Stata and Stata's response.
Command frame	You issue commands to Stata from this frame.

Variables frame	Every variable in the loaded data set is listed here.
Previous Commands frame	Every command you have issued to Stata during the session is stored here.

The "Previous commands" and "Variables" frames are both intended to make Stata easier/faster to use interactively since you can simply click on variable names or previous commands to load them into your command window instead of typing the entire command or list again.

Stata .do Files
Usually you need to issue many commands to Stata and you would like the ability to issue them again easily and/or document all of the commands you've issued. To do this, you need a "do" file. A do file is just a regular text document that you can create in a text editor or Stata's "do file editor" which contains all of your Stata commands. When you run a do file by selecting the "do" option from the file menu, Stata opens your do file and runs each line one at a time as if you had typed each line in the command frame.

3.3.3 Using .ado Files

The RDC servers are intentionally isolated from the web, which prohibits the standard Stata procedure for applying updates to the software. If you need an ado file that isn't available, retrieve it outside the RDC from the Stata web site, and submit it to your administrator as a user provided program. You can also check /apps/shared/stata to see if what you're looking for is already uploaded there. Once uploaded, to use an .ado file you can

(a) specify the full path when you call it from your do program (e.g., **/apps/shared/stata/<ado_file_name>**),
(b) copy it to your stata working directory where stata will see it (type adopath in Stata to see the order Stata looks for ado files), or
(c) set a system directory with the stata sysdir command and define its path. For example, **`sysdir set PLUS "/projects/programs/ado"`** will reassign PLUS (this usually points to ~/ado/plus/ which is not a directory in our systems) to your personal ado directory in your project space (you will need to create this directory). Type the command sysdir in stata to check that this worked.

3.3.4 System Performance for Stata Users

It is CRITICAL to manage memory when using Stata. Stata sessions require dedicated system memory allocated to them and this may cause the server to run out of system memory. Please use the following guidelines.

1) Do program development and debugging on a subset of data. Most interactive use of Stata can be performed with a subset of observations. The command to get a subset is **use <FileName> if _n<=200** (This uses only the first 200 observations of the file.)
2) Don't use the whole data set if you don't need to. Regularly, to open a Stata data set, you type **use data_set**. If you want to run analysis on specific variables in that data set, let's say variables named a, b, and c; then in Stata type **use a b c using data_set**.
3) Close out Stata to release memory. Memory is not released by Stata until the application itself is closed out, even if the program finishes running or if one resets memory with the set memory command. So, if you are not actively using Stata, then please close it.
4) Do not open multiple interactive sessions of Stata. (This follows from number 3.)
5) Run debugged Stata jobs in batch mode.
   `qstata program_name.do &`

6) Be courteous to your fellow users.

7) Kill off abandoned jobs. If you exit abnormally, then in many instances the system leaves your job(s) running or idling with the memory tied up. This strains the system and causes it to behave erratically. Use **`qdel <job_id>`** [ENTER] to delete jobs that are running or in the PBS Pro queue.

3.3.5 A Stata Example

Below is a very short Stata program that uses some basic Stata commands. Each line has a comment that starts with /* and ends with */ that explains what the code is doing.

log using "/projects/logs/program1.log"
/* By default, Stata output is sent to the output window and then discarded by Stata. If you want to save the results from a regression or to document what you're doing, then you will want to create a log file. A log file will take any output sent to the output window and store it in a file that you name. In this case, we've named both the file and the path; if you exclude the path, then your working directory is automatically selected. Also, note that we included the extension ".log" to our file name. If you leave that out, then Stata will by default create a ".smcl" file which is usually less desirable. */
use MyDataSet
/* Loads a data set called "MyDataSet.dta." We have left off specifying a path, so Stata will look for this file in the current working directory. Stata assumes the .dta. Therefore, if you have a Stata file called "MyDataSet.txt" then you will need to specify the ".txt" at the end. */
sum /* Gives you summary statistics for the variables that have been loaded. */
gen x2 = x*x
/* Creates a new variable named "x2." Stata will go through every observation and square the x value for that observation, storing the result in x2. Note, if x2 already exists then Stata will issue an error. */
tabstat x x2, stats(mean sd) column(stats)
/* Summary statistics for x and x2, same as **sum**marize command but excludes min and max which do not pass disclosure avoidance review (since min and max are statistics based on one observation, one survey respondent). */
edit /* Opens a spreadsheet like data editor. */
replace x2 = 0 if x == -5
/* Searches through every observation of x and if it finds one that equals -5, it replaces x2 with the value of zero. Note, in Stata you set values with one equal sign, and compare values with two equal signs. */
regress y x x2 /* Runs a simple OLS regression with y as the dependent variable. */
graph twoway scatter y x /* Graphs y and x. */
log close /* Closes your log (any further output will not be recorded). */
clear /* Removes all data from memory. */
exit /* Quits Stata. */

3.3.6 Printing Graphs to File

To export a graph from stata into an encapsulated postscript file, first generate the graph (using the "graph" command, or "graph use filename" if the graph has been previously saved), and then run the command "graph export filename.eps". The help file lists other supported formats for the command (e.g. tif). Note that some formats are not supported (e.g. pdf isn't supported by the translator, and wmf isn't supported in unix). Some researchers use .png files for LaTeX; graph export filename.png works in interactive stata but not in qstata (e.g. which calls stata-se and not xstata-se). You can still save stata graph files in batch, and then do the exports as a separate program.

If you want to interactively print to a postscript file in Stata from the print menu, File → print. Select "print to file" and then select where in your project directory you'd like to save it. Click 'Print." The command displayed in the Stata window should read something similar to:

```
translate @Graph /projects/data/Graph_name,
translator(Graph2ps)...
```

You will receive an error message. Copy the Stata command from the display window and place double quotes around the file path and file name as shown below and resubmit the command.

```
translate @Graph "/projects/data/Graph_name",
translator(Graph2ps)...
```

This will save the file "Graph_name" in the designated folder as a postscript file.

3.3.7 Stata Help

Each RDC has a set of Stata manuals. Outside of the RDC, you can access Stata online documentation. A couple good links are http://www.stata.com/links/resources1.html and http://www.stata.com/support/faqs/.

3.4 Gauss

To start an interactive Gauss session, at the prompt of a terminal window, type **`qsub -I -X`** [ENTER] to get to a research node, then type **`tgauss`** [ENTER] at the prompt of a Konsole window. Your shell will be occupied; open another shell in the same Konsole window if you need it. (From the menu Sessions → New Shell.)

Outside of the RDC, you can access Gauss online documentation. http://www.aptech.com/pdf_manuals.html and http://www.aae.wisc.edu/aae637/gausscode.htm

3.5 MATLAB

To launch a MATLAB batch job, create a batch script.

	Script text	Description
(1)	`#!/bin/bash`	(1) Specifies which kind of shell to use (bash) – never change this.
(2)	`#PBS -l select=1:ncpus=1:mem=8gb`	(2) Tells PBS Pro to expect directives and what they are. For clarity, the first item after #PBS is dash "L" and next is select equals one, ncpus is number of CPUs.
(3)	`cd /projects/programs`	(3) Change directory to full path of program file.
(4)	`matlab -nodisplay -nosplash -r program_name.m > program_name.log`	(4) Command to launch job. This line is recreated below to show where spaces are.

```
matlab -nodisplay -nosplash -r program_name.m > program_name.log
```

All users of the RDC share a limited number of MATLAB licenses. If you absolutely must run more than one MATLAB job at a time, then be sure to run both of them on the same node as the licenses are node and user specific. To do this, create your matlab job scripts as above – say they are called matlab1 and matlab2. To ensure that they run on the same node, add that requirement to your resource request list in the qsub command. For example:

qsub –l host=hpc-scompute1 matlab1
qsub –l host=hpc-scompute1 matlab2

will require that the jobs matlab1 and matlab2 be run on the host hpc-scompute1. If hpc-scompute1 is busy, the jobs will queue until it becomes available.

Outside of the RDC, you can access MATLAB online documentation.
http://www.mathworks.com/access/helpdesk/help/techdoc/index.html?/access/helpdesk/help/techdoc/

3.6 R

To launch a R batch job, at the terminal prompt type **`qR program_name`** [ENTER]

To start an interactive R session, at the prompt of a terminal window, type **`qsub -I -X`** [ENTER] to get to a research node, then type **`R`** [ENTER].

Be sure not to use lowercase "r" as this will give you an error message. Your shell will be occupied; open another shell in the same Konsole window if you need it. (From the menu Sessions → New Shell.)

Outside of the RDC, you can access R online documentation.
http://cran.r-project.org/doc/manuals/R-intro.html and http://cran.r-project.org/manuals.html

3.7 Sudaan

We offer SAS-callable Sudaan. Since SUDAAN is SAS-callable, you can simply add SUDAAN commands to your SAS program and submit the SAS program using the methods described in section 3.2, "SAS" (using qsas, using your own jobscript with qsub, or in an interactive batch session)

3.8 Other Thin Client Applications

3.8.1 Konsole

Konsole is a program that manages shells for you; each terminal session is given a tab. Some helpful commands for Konsole are

[CTRL]+c ends the command and returns to prompt. This is helpful if you submit a command and are getting *a lot* of results.

↑ and ↓ at the prompt, these browse the command history.

[SHIFT]+[PAGEUP] and [SHIFT]+[PAGEDOWN] allows you to brows the terminal buffer (to see the text that has scrolled off the screen).

[TAB][]TAB] (tab pressed quickly in succession) at the prompt will list all commands available on the system that fit the criterion you specify. For example, if you type **`b`** at the prompt and then press [TAB][TAB], all commands that begin with b will be shown.

Konsole will automatically open in your home directory. To help ease navigation of directories Konsole has a bookmark tool. You can use this instead of the often long **`cd`** command (**`cd /projects/data/…`**).

You can also change the default directory for the Konsole *button* that you created in section 1.2.1.1. Right click on it → Configure Terminal Button… → Application tab → in the field "Work path", type the directory you want opened by default (e.g., /projects/).

It is recommended that you change the size of the Konsole viewing window. Go to Settings → Size → 80X40(XTerm); then Settings → save as default.

Outside of the RDC, you can access Konsole online documentation.
http://docs.kde.org/development/en/kdebase-apps/konsole/index.html

3.8.2 Dolphin

Dolphin is the GUI file management application and it is a web browser. You can launch Dolphin
GUI: Click the "File Manager" icon in the Red Hat menu
CLI: At the prompt type **`dolphin &`**.

Note you may have to press [F9] in order to see the left-side navigation bar. To get to your home directory, click the House tab. To get to the root directory (the parent directory that lists all subordinate directories), click the red folder tab.

If you would like Dolphin to open in a specific directory, you can type the entire path name (i.e., **`dolphin /public/programs/sas &`**). There is no corollary function for GUI, but you can change the directory assignment for the Home button in the Dolphin window. From Dolphin's menu click Settings → Configure Dolphin… → in the field "Home URL:" type the directory you would like Dolphin to go to when you click Dolphin's home button.

Tip: you can open two Dolphin sessions to make drag-and-drop file management easier.

To use Dolphin as a web browser, simply type the address (**http://rdcdoc.ces.census.gov**) in the location bar and then bookmark it. (Remember you do not have regular access to the internet in the RDC. http://rdcdoc.ces.census.gov is our intranet site for help resources.)

Dolphin has tabbed browsing. If you do not see tabs go to Settings → configure Dolphin → Web Behavior → Tabbed browsing box → uncheck the box "hide the tab bar…".

Outside of the RDC, you can access Koqueror online documentation.
http://docs.kde.org/stable/en/kdebase-apps/dolphin/index.html

3.8.3 OpenOffice

OpenOffice.org is an open-source office software suite. To open OpenOffice.org…

GUI: Red Hat → Office → OpenOffice.org Writer (for the word processor) or
Red Hat → Office → OpenOffice.org Calc (for the spreadsheet)
CLI: at the prompt type **`oowriter &`** for the word processor and **`oocalc &`** for the spreadsheet.

If you wish to disclose a formatted document that you created using OpenOffice.org, you can either "save as" a Microsoft Office file extension, i.e., .doc or .xls, (and possibly encounter formatting errors) or you can download OpenOffice.org to your home computer for further editing. OpenOffice.org is a free program.

Outside of the RDC, you can access OpenOffice online documentation.
http://documentation.openoffice.org/

If you have a very old project, you may have OpenOffice documents in an old format. If so, you might have to save them in the new format since the OpenOffice.org suite has been updated to a version which is international open standard format compliant. This means that documents that were created with the older version of the suite need to be saved in the new format to function properly (e.g., print). Simply click on your old file to open it, from the menu file → save as, when the new window pops up use the following table to determine your "save as" file extension.

Type of Document	old file extension	save as…
Text	.sxw	.odt
Spreadsheet	.sxc	.ods

3.8.4 Text Editors

We offer a number of text editors on the thin client system. These include kate, kedit, kwrite, and emacs. You can launch most of them from Red Hat → Utilities → Editors.

3.8.4.1 Kate

We recommend that you use the text editor that you are comfortable with; however, if you are not committed to one or the other, then we recommend Kate for the following reasons.

- Kate allows you to select columns of data which makes manipulation of files easier. This is especially helpful for editing Stata log files by letting you easily delete undisclosable columns of output. To toggle this function on/off either use the mouse at the menu then Edit → Block Selection Mode or the keyboard combination [CTRL]+[SHIFT]+B.
- When using Kate to monitor batch processes, you can refresh the screen when your program has written to the file (as compared to opening and closing the file to determine if additional activity has occurred). Press F5.
- Kate is quite intuitive, behaving rather like a word processor in many ways, it has good help, and allows you to assign "sessions" which allows you to organize your programs. A Kate session may become cluttered if you do not normally sort your documents into different sessions. To remove a file from an active session in Kate, highlight it (on the right-side navigation bar) and then click the circle icon with an “X” (this is next to the printer icon).
- You can request that Kate recognize the language of many different applications and Kate will highlight the document accordingly.
  - For Matlab: Tools → Highlighting → Scientific → Matlab
  - For SAS: Tools → Highlighting → Sources → C++ (This is imperfect, but works well enough in most circumstances.)
  - For LaTeX: Tools → Highlighting → Markup → LaTeX
  - For script files: Kate automatically highlights script files when the file extension is bash.

Tip: if you run a search in Kate, then you press F3 to continue to the next incidence of the search term.

3.8.5 Pager

Pager is a handy tool that manages the four desktops that are available to you. On the panel at the bottom of your desktop you will see a little table with quadrants numbered 1 through 4. You can click on the numbers to get to a different desktop. The different desktops are nice so that no one becomes too cluttered. You may like to make the desktops task specific (e.g., by node, or application, etc.).

4 Trouble Shooting

If you are experiencing computer troubles, then these are the things to try and/or check before notifying your Administrator. Once you have located the description of your problem, it is best to follow the remedial steps in the order given.

4.1 The Description of the Problem is…

4.1.1 An Authentication Error

The problem is that the system cannot confirm your authority to log in.

1) Check your username and password.
2) Reboot the client. Do NOT just press the power button to reboot the thin client! Click "Cancel" to return to the launch menu. From there, click client → restart. After reboot, attempt log in again.
3) If these do not correct the problem, then notify your Administrator. The problem is with your account, which can be due to many things (e.g., the number of failed log in attempts, an expired password, etc.). There is nothing that you can do on your own to solve these.

4.1.2 A Connection Time-out

The problem is that the thin client is not communicating with the server.

1) Make sure the data cable is connected properly to the thin client. If it was attached improperly, then try to log in again.
2) If the cable was fine or if the second log in attempt also failed, reboot the client.
3) Try a different thin client.
4) Notify your Administrator.

4.1.3 Session Terminates

Although this usually happens when there is a significant problem with the servers, sometimes the system hiccups and sessions are terminated for no real reason.

1) Reboot the client and attempt log in again. If asked if you would like to resume your previous session, say yes. If this does not work, then
2) Return to the sign-in desktop and try again; but if given the option, then *terminate* the previous session.
3) Notify your Administrator.

4.2 Reporting Problems

Any time you have a problem with the Thin Clients, please notify your Administrator. The more detail you can give them about what went wrong, how you tried to fix it, whether you were successful or not, the more likely it is they will be able to help you fix it. You should **always** provide your Administrator with the following information.

1) Date and time the problem began.
2) The node(s) you were using.
3) Your User ID.
4) If it is a PBS Pro job, what is the job id.
5) A detailed description of the problem, including what applications and/or data sets you were using, what you were doing when the problem occurred. The exact wording of the error message is *very* helpful. The more detailed the description, the better and more quickly Census computer technician(s) will be able to fix the problem.

6) Description of any remedial steps you took to diagnose and solve the problem.

If you are unable to contact your local RDC Administrator in a reasonable amount of time, you may try contacting the Administrator at a different RDC. You can find contact information at http://www.census.gov/ces/main/contact.html, but you will need to leave the RDC to access this information because devices with internet connectivity are prohibited in the RDC. All computer problems should be reported to a RDC Administrator or other CES employee.