

Stata commands currently supported by ACRO

We focus on Stata V17 and above as the syntax changed significantly from then, but support for Stata V16 and below is broadly the same.

This is an ongoing project we want to be driven by researchers' needs

So **please let us know** if there are commands or things you would like us to support by:

- emailing sacro.contact@uwe.ac.uk, or
- adding an issue to github.com/ai-sdc/acro

Author james.smith@uwe.ac.uk, **Last Updated** 26 February 2026

Introduction and General Concepts

Aim

ACRO is designed to provide drop-in replacements for commonly-used Stata commands that add functionality to let researchers and TRE staff fulfill their collective duty to produce *Safe Outputs*.

Most data management is done exactly as before, the difference comes when you issue a *query* that produces an output you may want to take out of the TRE.

ACRO provides automated checks against risks you should be checking e.g.: *small group sizes*, *class disclosure*, *dominance* and *low residual degrees of freedom*.

Researchers are provided with this feedback immediately so they have the power to choose between: - **redesigning** their outputs to reduce risks, - **applying a mitigation strategy** (e.g. *acro* can provide automated suppression of table cells with low counts and adjust row/column totals), - **requesting an exception** to strict rules-based checking.

ACRO *sessions*

The key concept is that of an *acro session*: comparable to doing a day's work and requesting some outputs at the end. - Of course you can save your work at any stage using normal Stata functionality. - We anticipate that people may want to do preparatory work, and then add *acro* prefixes to queries, and session commands to their code and do a final run through.

Caveat

At present most *acro*-assessed outputs are saved as csv files or image files.

- So **we currently assume you will format your results later** e.g. once they have been released from the TRE and are producing your publication.
- We have open issues to support more formatted work - and welcome feedback on that.

Simple worked example

This uses an open-source data set about nursery admissions

Start Analysis session

```
acro init
```

Load and manipulate data

```
use "../data/nursery_dataset" ACRO is not involved in how you do this.
```

Make some tables

using the standard Stata `table` commands prefixed by `acro`, for example:

```
acro table parents finance In this example an output is added to the acro session with the feedback that it passes all the relevant checks
```

```
acro table parents recommend In this example the feedback attached to the output is that it fails some disclosure checks.
```

You can either:

- recode the data if you want: this is manipulation and up to you
- or make a request to override the standard rules:

```
acro add_exception "output_1" "Add your reason here"      acro reports the id of each output when they are created.
```

- or ask `acro` to automatically suppress any disclosive cells
 - and update row/column totals if needed
 - and add an exception message messaged saying it has done that.

```
acro enable_suppression
```

```
acro table parents recommend
```

Tidy up and finish the session

```
acro print_outputs
```

```
acro rename_output "output_2" "suppressed table parents by recommend"
```

```
acro add_comments "output_0" "table of finance by parents"
```

```
acro remove_output "output_1"
```

```
acro finalise "my new stata outputs folder"
```

ACRO session management commands

`acro init, config(filename)`: starts an *acro* session using a TRE-provided risk appetite in *filename.yaml*

`acro custom_output myfilename.xxx`: adds a custom output i.e. code, or a paper version, or images of plots or results from analyses we can't auto-check (yet)

`acro enable_suppression`: turns on automatic suppression for subsequent outputs

`acro disable_suppression`: turns off automatic suppression for subsequent outputs.

`acro print_outputs`: prints list to screen of outputs in current session and their status

`acro remove_output output_id`: removes a named output from an *acro* session

`acro rename_output output_id new_name`: renames a named output from an *acro* session

`acro add_comments output_id comment_string`: adds a comment (string) to a named output

`acro add_exception output_id exception_string`: adds an exception request (string) comment to a named output

`acro finalise [output_dir] [filetype]`: __wraps up the session, writing the outputs to a named directory (default= "stata_output") in json (default) or xlsx format.

Note that *acro* will not overwrite your work

This means the name of the directory you pass to *finalise* must not exist.

You can: delete old versions manually; provide a new name; or create the name in code.

The file `acro_demo_2026.do` (and pdf versions) show how to add the current date and time to folder names in Stata.

ACRO commands for creating tables

For Stata versions V17 and beyond you can make tables of frequencies (or other statistics) using Stata's *table* command.

`acro table (rowvars) (colvars) (tabvars) [if] [in] [,options]`

See Stata's built-in help for details of how to use this. - You can currently report one or more of the following statistics: *count* (default, same as frequency), *mean*, *median*, *sum*, *std*, and *mode*.

There are some minor differences, which will be addressed in the next version.

Currently not supported are:

- **weights within the options**
- **formatting commands (see above).**
- compound manipulation/query commands such as `xi:` that create indicator variables from categorical ones on-the-fly.

Workaround for xi

```
xi: regress myvar1 i.myvar2 myvar3
```

Can be broken down into two stages using Stata commands:

```
xi i.myvar2 creates the binary variables
```

```
regress myvar1 `ds _Imyvar2*' myvar3
```

ACRO commands for creating regression models

```
acro regress dependent_var independent_vars [if] [in]: linear regression
```

```
acro logit dependent_var independent_vars [if] [in]: logistic regression
```

```
acro probit dependent_var independent_vars [if] [in]: probit regression
```