dscheck

August 16, 2005

Abstract

Table-driven check of dataset contents

1 Instruments/Modes

Instrument	Mode

2 Use

pipeline processing	yes
interactive analysis	no

3 Description

This task performs a range of low-level checks on the contents of datasets (FITS files). Examples are:

- Ensuring a FITS file contains an extension of a given name
- Ensuring a FITS HDU (primary or extension) contains a keyword of a given name
- Ensuring a FITS binary table contains a column of a given name
- Checking that a FITS keyword falls in a specified range
- Checking that the values in a column of a FITS table fall within a specified range

The checking process it table-driven, ie. the checks to be performed are defined in an ASCII file. The format of this file is best demonstrated by example:

```
   # This is a comment
1  EXISTS PRIMARY.CONTENT          Check existence of a keyword in primary header
                                   (Blank lines are ignored)
2  EXISTS SRCLIST                  Check existence of extension named SRCLIST
3  EXISTS SRCLIST.TESTKEY          Check existence of keyword in extension named SRCLIST
4  EXISTS SRCLIST.SRC_NUM[]        Check existence of column in a table named SRCLIST

5  RANGE PRIMARY.RA_PNT 0 360      Check range of keyword in primary header
6  RANGE PRIMARY.DEC_PNT -90 90

7  RANGE SRCLIST.TESTKEY -500 100  Check range of keyword in extension named SRCLIST
8  RANGE SRCLIST.SCTS[] 0 1000     Check range of values in column SCTS of table named SRCLIST
```

There is some implicit checking performed behind the scenes, eg. checking the range of a keyword implicitly checks for it's existence, and checking the existence of a keyword implicitly checks for the existence of the header it is in. Thus in the table above check 3 implies check 2, and check 7 implies check 3 which implies check 2 (ie. checks 2 and 3 are redundant).

When anomalies are found they are written to `STDOUT`. All checks will be performed before the task exits. If any checks fail the task will exit with an error, otherwise it will exit normally (this is the mechanism for communicating with the pipeline control system).

4 Parameters

This section documents the parameters recognized by this task (if any).

Parameter	Mand	Type	Default	Constraints
set	yes	dataset		

Defines the dataset to be checked

checkfile	yes	filename		

Name of the file defining the checks to be performed

5 Errors

This section documents warnings and errors generated by this task (if any). Note that warnings and errors can also be generated in the SAS infrastructure libraries, in which case they would not be documented here. Refer to the index of all errors and warnings available in the HTML version of the SAS documentation.

primaryColumn *(fatal)*
Attempt to refer to a column in the FITS primary extension

6 Input Files

1. Dataset to be checks
2. Check file defining checks to be performed

7 Output Files

1.

8 Algorithm

```
   subroutine dscheck

open dataset
read check file

foreach (check)
do check
end

print output

raise error if any check fails

   end subroutine  dscheck
```

9 Comments

•

10 Future developments

- Currently range checks only really make sense for numeric keywords and table columns. The task doesn't do any checking to verify that the keyword/column being checked is in fact numeric, and almost certainly it should do.
- Limits for range checks are restricted to numeric constants. It would be possible to extend the checks to be arithmetic expressions, etc.
- The task does not do much checking on the validity of the check file (eg. invalid ranges like $200 < x < 100$ are not trapped).
- Further checking functions can be added quite simply, eg. a function *name* can be implemented by adding a perl subroutine called `check_`*name* to the `dscheck.pl` script.

References