STAF Desk Reference

Generated by stafRef.c on Jul 09 1998

---

0. Introduction

There is an index at the end.

If you have corrections please write:
`ward@physics.utexas.edu`.

---

1. AMI, Analysis Module Interface

1.1 /AMI/CALL PAM [TABLES]

	name	description	type	default/range
required	PAM	Physics Analysis Module function name.	character string	no default
optional	TABLES	List of PAM argument tables.	character string	D='-'

OBSOLETE – PLEASE USE AMI/MODULE/CALL.

1.2 /AMI/COUNT

Show the current count of AMI worker objects.

DESCRIPTION:

COUNT is a readonly long attribute which reflects the number of AMI worker objects currently registered with the AMI object factory. Constructing a new AMI worker object increments COUNT by 1, destroying an existing AMI worker object decrements COUNT by 1.

AMI worker objects include:
amiInvoker - See AMI/MODULE
- An object for invoking user written analysis module functions.

ARGUMENTS:

None.

RETURN:

The current value of COUNT is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current count of AMI worker objects.

StAF AMI/COUNT
AMI: Object count = 5

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

1.3 /AMI/LIST

List all currently registered AMI worker objects.

DESCRIPTION:

Show a one-line description for each AMI worker object currently registered with the AMI object factory in a table format.

The one-line description for each object is the result of an invokation of that object's listing method. The typical content of this listing is:
0 OID
The object's OID attribute (see SOC) presented as "%5d".
1 Lock State
The object's LOCK attribute (see SOC) presented as the
divider character between the OID column and the NAME:OBJECT
column. An object whose LOCK attribute is TRUE (cannot be
deleted) uses the "-" character, whereas an object whose
LOCK attribute is FALSE (can be deleted) uses "—" character.
2 NAME:OBJECT
The object's NAME attribute (see SOC) presented as "%-15s".
Object names longer than 15 characters are abreviated with a
" " character at midpoint.
An object name is synonymous with an object instance.
3 TYPE:CLASS
The object's TYPE attribute (see SOC) presented as "%-15s".
Object types longer than 15 characters are abreviated with a
" " character at midpoint.
An object type is synonymous with an object class.
4 DESCRIPTION
A class-specific description of the object.
For AMI objects the number of table arguments is listed.

AMI worker objects include:
amiInvoker - See AMI/MODULE
- An object for invoking user written analysis module functions.

ARGUMENTS:

None.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
amiFactory::list()
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. List all current AMI worker objects.

staf++ ami/list
+————————————————————

+——-+—————+—————+————————-
— IDREF — NAME:OBJECT — TYPE:CLASS — DESCRIPTION
+——-+—————+—————+————————-
— 9 — tcl_mak lusters — amiInvoker — 6 arg.s
— 10 — tpeam — amiInvoker — 13 arg.s
— 11 — tpham — amiInvoker — 10 arg.s
— 12 — reformat — amiInvoker — 13 arg.s
— 13 — tfc_calc_delta — amiInvoker — 2 arg.s
— 14 — tfc_stability — amiInvoker — 4 arg.s
— 15 — tstam — amiInvoker — 8 arg.s
— 16 — tstgain — amiInvoker — 5 arg.s
— 17 — xyz — amiInvoker — 5 arg.s
— 18 — tpg_main — amiInvoker — 3 arg.s
— 19 — tpt — amiInvoker — 4 arg.s
— 20 — tpt_sts — amiInvoker — 6 arg.s
+——-+—————+—————+————————-

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

1.4 /AMI/MODULE/CALL SOREF [TABLES]

	name	description	type	default/range
required	SOREF	amiModule object SORef	character string	no default
optional	TABLES	List of PAM argument tables.	character string	D='-'

The way to get work done. CALL executes the PAM (SOREF) and specifies what tables it will operate on.

DESCRIPTION:

CALL is a member function of objects which implement the amiModule interface.

Invoke a Physics Analysis Module on data tables. IN and INOUT tables must exist before being passed to the PAM. OUT tables can, but need not, exist. Non-existent OUT tables will be created at invokation of the PAM by CALL.

Non-existent OUT tables can be created with a user-specified memory allocation by specifying the number of rows to be allocated in parenthesis after the table name (The default is to allocate only one row's worth of memory.).

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the amiModule interface.

TABLES - List of PAM argument tables.
- A list of memory-resident data tables, by name, upon which to invoke the Physics Analysis Module function.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
amiModule::CALL
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Call an example analysis module function.

Kuip AMI/MODULE/CALL pam tab1 tab2(4000) tab3
If tab1 exists, and tab2 and tab3 are output tables, this command
will create tab2 with 4000 rows allocated and tab3 with 1 row
allocated, and then call pam on the three tables.

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the amiModule interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

1.5 /AMI/MODULE/RANK SOREF

	name	description	type	default/range
required	SOREF	amiModule object SORef	character string	no default

Get the RANK of a Module. I.e. the number of call arguments in its Module definition .idl file.

DESCRIPTION:

RANK is a readonly attribute which reflects the value of the RANK
attribute of the amiModule SOREF. Readonly attributes cannot be changed
from the user interface.

NB. Readonly attributes are not necessarily static attributes.

The rank of an analysis module function is determined by the IDL
(Interface Definition Language) definition of the analysis module
function interface. In short, the RANK is the number of tables in
the PAM's call list.

The IDL file:

```
/* connect_the_dots.idl */
#include "PAM.idl" // generic include
#include "point.idl" // point table type definition
#include "line.idl" // line table type definition
interface connect_the_dots : amiModule
STAFCV_T call (
in point beg_pts, // input table of type point
in point end_pts, // input table of type point
out line lines // output table of type line
);
;
```

and outputs one table of type line. This analysis module has a rank of three (i.e. It takes three tables as arguments.).

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the amiModule interface.

RETURN:

The current value of RANK is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current value of the RANK attribute of amiModule "pamf".

Kuip AMI/MODULE/RANK pamf
AMI: Analysis module rank = 2

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the amiModule interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

1.6 /AMI/MODULE/SHOW SOREF

	name	description	type	default/range
required	SOREF	amiModule object SORef	character string	no default

Show definition of Analysis Module invoker. I.e. show the tables expected in calling the PAM.

DESCRIPTION:

SHOW is a member function of objects which implement the amiModule interface.

The definition of a PAM shows the number, types, and I/O modes of the data tables upon which the PAM operates.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the amiModule interface.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
amiModule::SHOW
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Show the definition of amiModule "pamf".

Kuip AMI/MODULE/SHOW pamf
AMI: Table Specification = ...
struct scalars
short aShort;
unsigned short aUshort;
long aLong;
unsigned long aUlong;
char aChar;
octet aOctet;
float aFloat;
double aDouble;
; .
AMI: Table Specification = ...
struct vectors
short bShorts[3];
unsigned short bUshorts[3];
long bLongs[3];
unsigned long bUlongs[3];
char bChars[3];
octet bOctets[3];
float bFloats[3];
double bDoubles[3];
; .

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the amiModule interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

---

2. ASU, Analysis Service Utility

2.1 /ASU/DATE

Print the current date and time to stdout.

DESCRIPTION:

DATE prints to stdout a character string showing the current date and time.

ARGUMENTS:

None.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the

DATE
function is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Print the current date and time.

Kuip ASU/DATE
ASU: Date = Tue Dec 16 10:28:25 1997

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

2.2 /ASU/EML/BEEP_ON_ERROR [BEEP]

	name	description	type	default/range
optional	BEEP	Either TRUE, FALSE, or SHOW.	character string	D='SHOW'

You can type ON/OFF instead of TRUE/FALSE.

SHOW reports the current value.
The other possible values are self-explanatory.

DESCRIPTION:

Self-explanatory.

Beeping is ON by default when Staf starts.

ARGUMENTS:

BEEP - Either ON, OFF, or SHOW.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
asu_beep
function is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

Kuip ASU/EML/BEEP_ON_ERROR ON

EXCEPTIONS:

BUGS:

None.

SEE ALSO:

2.3 /ASU/EML/DEMAND_ERROR_ACKNOWLEDGEMENT [DEMAND]

	name	description	type	default/range
optional	DEMAND	Either TRUE, FALSE, or SHOW.	character string	D='SHOW'

You can type ON/OFF instead of TRUE/FALSE.

SHOW reports the current value.
The other possible values are self-explanatory.

DESCRIPTION:

The idea is to keep error messages from scrolling off
the top of your screen unnoticed.

However, if you have many benign errors, you will probably
want to turn DEMAND_ERROR_ACKNOWLEDGEMENT off.

This feature is OFF by default when Staf starts, for
backward compatibility.

ARGUMENTS:

DEMAND - Either ON, OFF, or SHOW.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
asu_demand_ack
function is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

Kuip ASU/EML/DEMAND_ERROR_ACKNOWLEDGEMENT OFF

EXCEPTIONS:

BUGS:

None.

SEE ALSO:

2.4 /ASU/EML/PRETTY_FORMATTING [PRETTY]

	name	description	type	default/range
optional	PRETTY	Either TRUE, FALSE, or SHOW.	character string	D='SHOW'

SHOW reports the current value.
The other possible values are self-explanatory.

DESCRIPTION:

You can type ON/OFF instead of TRUE/FALSE.

Prettification formats error messages for readability.
It does not remove any information.

Pretty error messages are ON when Staf starts.
If you suspect a bug in the prettification mechanism, you
can try turning it off.

ARGUMENTS:

PRETTY - Either ON, OFF, or SHOW.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
asu_pretty
function is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

Kuip ASU/EML/PRETTY_FORMATTING
ASU: Error messaging prettification is ON.
Kuip ASU/EML/PRETTY_FORMATTING OFF
Kuip ASU/EML/PRETTY_FORMATTING
ASU: Error messaging prettification is OFF.

EXCEPTIONS:

BUGS:

None.

SEE ALSO:

2.5 /ASU/FFLUSH

Flush the print buffers of all open streams.

DESCRIPTION:

FFLUSH calls fflush(0);

This allows the stderr and stdout buffers to be explicitly flushed.

ARGUMENTS:

None.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
FFLUSH
function is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. More guidance needed here.

StAF ASU/FFLUSH

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

2.6 /ASU/HELLO MESSAGE

	name	description	type	default/range
required	MESSAGE	Salutory message.	character string	D=' '

Print a salutory message to stdout.

DESCRIPTION:

HELLO is a useful command for testing that the KUIP interface is
working properly. It has no side effects and no interesting or
comlicated code.

ARGUMENTS:

MESSAGE - Salutory message.
- Any printable character string message to be printed to stdout.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
HELLO
function is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Print a typical salutation.

Kuip ASU/HELLO Bill
ASU: Hello, Bill

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

2.7 /ASU/MALLOC/LEVEL [NEW_VALUE]

	name	description	type	default/range
optional	NEW_VALUE	Memory allocation debug level	integer	D=0 R='0,2'

Get or set the "asuAlloc.h" debug level.

DESCRIPTION:

LEVEL is a read-writable parameter which determines the behavior of functions in the asuAlloc.h package.

To get the current value of LEVEL, leaving LEVEL unchanged, do not specify a new value in the optional argument NEW_VALUE.

To set a new value of LEVEL, specify the new value as the optional argument NEW_VALUE.

ARGUMENTS:

NEW_VALUE - Memory allocation debug level.
- 0 Print current level.
- 1 = FAST Only call normal malloc & free...
- 2 = COUNT ...and count calls to malloc & free...
- 3 = TRACE ...and keep trace of memory locations...
- 4 = FILL ...and fill allocated memory w/ pattern...
- 5 = VERBOSE ...and print a message every time.
- DEFAULT: Show the current value of LEVEL, do not change it.

RETURN:

The current value of LEVEL is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current MALLOC debug level.

Kuip ASU/MALLOC/LEVEL
ASU/MALLOC/LEVEL = ASU_MALLOC_FAST

EG2. Set the MALLOC debug level to COUNT and show current level.

Kuip ASU/MALLOC/LEVEL 2
Kuip ASU/MALLOC/LEVEL
ASU/MALLOC/LEVEL = ASU_MALLOC_COUNT

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the asuMalloc interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

2.8 /ASU/MALLOC/STATS

Print memory allocation statistics.

DESCRIPTION:

ASU/MALLOC/STATS prints to stdout the current statistics accrued by the "asuAlloc.h" function.

ARGUMENTS:

None.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
STATS
function is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Print memory allocation statistics before any work.

Kuip ASU/MALLOC/STATS
ASU_MALLOC: Memory allocation statistics:
mallocCalls 0, freeCalls 0, diff 0
mallocSize 0, freeSize 0, diff 0
asuMallocSize 0

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the asuMalloc interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:
ASU/MALLOC/LEVEL

2.9 /ASU/TIME

Show a time increment since t0.

DESCRIPTION:

TIME prints to stdout the time in seconds since the first time ASU/TIME was called in this process.

TIME provides a convenient tool for timing execution within StAF.

More guidance needed here.

ARGUMENTS:

None.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
TIME
function is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Time a WAIT statement.

Kuip ASU/TIME; WAIT ... 2; ASU/TIME
ASU: Time = 0.000000
...
ASU: Time = 1.994348

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

---

3. DIO, Data Input/Output

3.1 /DIO/COUNT

Show the current count of DIO worker objects.

DESCRIPTION:

COUNT is a readonly long attribute which reflects the number of DIO worker objects currently registered with the DIO object factory. Constructing a new DIO worker object increments COUNT by 1, destroying an existing DIO worker object decrements COUNT by 1.

DIO worker objects include:
dioStream - See DIO/STREAM
- A generic data stream object. Abstract base class for all dio stream objects.

dioFileStream - See DIO/FILESTREAM
- A data stream object associated with a disk file.
dioSockStream - See DIO/SOCKSTREAM
- A data stream object associated with a TCP/IP socket.
dioTapeStream - See DIO/TAPESTREAM
- A data stream object associated with a magnetic tape device.
- NOTICE – dioTapeStream is not yet implemented.

ARGUMENTS:

None.

RETURN:

The current value of COUNT is pushed onto the STAF_RESULT stack
(see SOC).

EXAMPLES:

EG1. Show the current count of DIO worker objects.

StAF DIO/COUNT
DIO: Object count = 18

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

3.2 /DIO/FILESTREAM/FILENAME SOREF

	name	description	type	default/range
required	SOREF	dioFilestream object SORef	character string	no default

Get the FILENAME attribute of the dioFilestream SOREF.

DESCRIPTION:

FILENAME is the XDF file on disk from which dioFilestream object SOREF
reads or to which dioFilestream object SOREF writes.
Readonly attributes cannot be changed from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the dioFilestream interface.

RETURN:

None.

EXAMPLES:

EG1.

staf++ dio/filestream/filename DST
DIO: File name = (/star/sol/users/love/data/dst1.xdf)

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the dioFilestream interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

3.3 /DIO/LIST

List all currently registered DIO worker objects.

DESCRIPTION:

Show a one-line description for each DIO worker object currently
registered with the DIO object factory in a table for quick,
simple perusal.

The one-line description for each object is the result of an invokation
of that object's listing method. The typical content of this listing is:
0 OID
The object's OID attribute (see SOC) presented as ”%5d”.
1 Lock State
The object's LOCK attribute (see SOC) presented as the
divider character between the OID column and the NAME:OBJECT
column. An object whose LOCK attribute is TRUE (cannot be
deleted) uses the ”-” character, whereas an object whose
LOCK attribute is FALSE (can be deleted) uses ”—” character.
2 NAME:OBJECT
The object's NAME attribute (see SOC) presented as ”%-15s”.
Object names longer than 15 characters are abreviated with a
” ” character at midpoint.
An object name is synonymous with an object instance.
3 TYPE:CLASS
The object's TYPE attribute (see SOC) presented as ”%-15s”.
Object types longer than 15 characters are abreviated with a
” ” character at midpoint.
An object type is synonymous with an object class.
4 DESCRIPTION
A class-specific description of the object.
For Filestreams this is the name of the file with indicators
whether it is read/write and whether it is open/closed.

ARGUMENTS:

None.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
dioFactory::list()
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. List all currently registered DIO worker objects.

staf++ dio/list
+———————————————————————————
+——-+—————+—————+——————————-
— IDREF — NAME:OBJECT — TYPE:CLASS — DESCRIPTION
+——-+—————+—————+——————————-
— 41 — GeoTables — dioFileStream — (R,C) /afs/rhic/star/starli
— 69 — raw_data — dioFileStream — (R,O) /star/mds/data/SD97/c
+——-+—————+—————+——————————-

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

3.4 /DIO/NEWFILESTREAM NAME FILE [MODE]

	name	description	type	default/range
required	NAME	Name for new dioFilestream object	character string	no default
required	FILE	File name of XDF data file	character string	no default
optional	MODE	Read/write mode	character string	D='R' R='R,W'

Create a new dioFilestream object.

DESCRIPTION:

Each dioFilestream created by the dioFactory shows up as an object managed by the dioFactory (see DIO/COUNT and DIO/LIST) and registered with the socCatalog (see SOC/COUNT and SOC/LIST).

ARGUMENTS:

NAME - Case-sensitive alphanumeric name for new dioFilestream object.
- Use this name as part of SOREF (see SOC) to specify this particular dioFilestream object in subsequent commands.

FILE - File name of XDF data file.
- Unix file name.

MODE - Read/write mode
- MODE = R - Read Only
W - Write Only

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
dioFactory::newFilestream
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Open an output file

STAF DIO/NEWFILESTREAM DST /star/sol/users/love/data/dst1.xdf W

EXCEPTIONS:

OBJECT_NOT_CREATED - The object creation failed. See error stack for detailed explanation of failure.

BUGS:

None known.

SEE ALSO:

DIO/FILESTREAM

3.5 /DIO/NEWSOCKSTREAM NAME HOST PORT [MODE]

	name	description	type	default/range
required	NAME	Name for new dioSockstream object	character string	no default
required	HOST	Host name of remote host	character string	no default
required	PORT	Socket port number	integer	R='1024:9999'
optional	MODE	Read/write mode	character string	D='R' R='R,W'

Create a new dioSockstream object.

DESCRIPTION:

Each dioSockstream created by the dioFactory shows up as an object
managed by the dioFactory (see DIO/COUNT and DIO/LIST) and
registered with the socCatalog (see SOC/COUNT and SOC/LIST).

ARGUMENTS:

NAME - Case-sensitive alphanumeric name for new dioSockstream object.
- Use this name as part of SOREF (see SOC) to specify this particular
dioSockstream object in subsequent commands.
- More guidance needed here.

HOST - Host name of remote host.
- or TCP/IP address of the host to which to connect.

PORT - Socket port number.
- Service port number to which to connect.
- PORT is ignored for MODE == W.

MODE - Read/write mode
- MODE = R - Read Only
W - Write Only

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
dioFactory::newSockstream
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. More guidance needed here.

EXCEPTIONS:

OBJECT_NOT_CREATED - The object creation failed. See error stack for
detailed explanation of failure.

BUGS:

None known.

SEE ALSO:

DIO/SOCKSTREAM

3.6 /DIO/SOCKSTREAM/HOST SOREF

	name	description	type	default/range
required	SOREF	dioSockstream object SORef	character string	no default

Get the HOST attribute of the dioSockstream SOREF.

DESCRIPTION:

HOST is a readonly attribute which defines the remote host to which to connect.
Readonly attributes cannot be changed from the user interface.

The HOST attribute has no meaning for a source dioSockStream object (ie. MODE = W).

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the dioSockstream interface.

RETURN:

None.

EXAMPLES:

EG1. Show the current value of the HOST attribute of dioSockstream object "bob".

StAF DIO/SOCKSTREAM/HOST bob
More guidance needed here.

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the dioSockstream interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.
The HOST attribute has no meaning if MODE = W.

SEE ALSO:

3.7 /DIO/SOCKSTREAM/MAXHANDSHAKES SOREF [NEW_VALUE]

	name	description	type	default/range
required	SOREF	dioSockstream object SORef	character string	no default
optional	NEW_VALUE	New value of MAXHANDSHAKES attribute	integer	D=-1 R='-1:10000'

Get or set the MAXHANDSHAKES attribute of the dioSockstream SOREF.

DESCRIPTION:

MAXHANDSHAKES is a read-writable attribute which determines how many times to attempt to establish a connection between dioSockstream object SOREF and the remote socket before failing.

To get the current value of MAXHANDSHAKES, leaving MAXHANDSHAKES unchanged, do not specify a new value in the optional argument NEW_VALUE.

To set a new value of MAXHANDSHAKES, specify the new value as the optional argument NEW_VALUE.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the dioSockstream interface.

NEW_VALUE - New value for the MAXHANDSHAKES attribute.
- DEFAULT: Show the current value of MAXHANDSHAKES, do not change it.

RETURN:

The current value of MAXHANDSHAKES is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current value of the MAXHANDSHAKES attribute of dioSockstream object "bob".

StAF DIO/SOCKSTREAM/MAXHANDSHAKES bob
More guidance needed here.

EG2. Set the MAXHANDSHAKES attribute of dioSockstream object "bob" to 123.

StAF DIO/SOCKSTREAM/MAXHANDSHAKES bob 123
More guidance needed here.

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the dioSockstream interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

N.B.- The server and client handshake protocols must match.

SEE ALSO:

3.8 /DIO/SOCKSTREAM/PORT SOREF

	name	description	type	default/range
required	SOREF	dioSockstream object SORef	character string	no default

Get the PORT attribute of the dioSockstream SOREF.

DESCRIPTION:

PORT is a readonly attribute which defines the service port on the

remote node to which to connect.
Readonly attributes cannot be changed from the user interface.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the dioSockstream interface.

RETURN:

The current value of PORT is pushed onto the STAF_RESULT stack
(see SOC).

EXAMPLES:

EG1. Show the current value of the PORT attribute of
dioSockstream "bob".

StAF DIO/SOCKSTREAM/PORT bob
More guidance needed here.

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the dioSockstream interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

N.B. PORT must be the number of a free port. See man socket.

SEE ALSO:

3.9 /DIO/STREAM/CLOSE SOREF

	name	description	type	default/range
required	SOREF	dioStream object SORef	character string	no default

Terminate communication with associated data stream. The state attribute
of the stream becomes "CLOSED".

DESCRIPTION:

CLOSE is a member function of objects which implement the dioStream
interface, including:

dioFileStream - See DIO/FILESTREAM
- A data stream object associated with a disk file.
dioSockStream - See DIO/SOCKSTREAM
- A data stream object associated with a TCP/IP socket.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the dioStream interface.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
dioStream::CLOSE

method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Close the "DST" Filestream.
staf++ dio/stream/close DST
staf++ dio/stream/state DST
DIO: Stream state = (CLOSED)

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the dioStream interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

3.10 /DIO/STREAM/GETEVENT SOREF [DATASET]

	name	description	type	default/range
required	SOREF	dioStream object SORef	character string	no default
optional	DATASET	In memory dataset name	character string	D='.'

Read a dataset from an XDF data stream into memory.

DESCRIPTION:

GETEVENT is a member function of objects which implement the dioStream
interface, including:

dioFileStream - See DIO/FILESTREAM
- A data stream object associated with a disk file.
dioSockStream - See DIO/SOCKSTREAM
- A data stream object associated with a TCP/IP socket.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the dioStream interface.

DATASET - Optional in-memory dataset name. If not given, the
dataset name found on the file will be used. If the dataset exists,
it will be replaced with the dataset from the stream, if not it will
be created. The stream must be open and in READ mode.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
dioStream::GETEVENT
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Read in an event from the stream "raw_data"

staf++ dio/stream/getevent raw_data

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the dioStream interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)
BAD_MODE_OR_STATE - Attempt to read from a writeonly stream or a closed stream.

BUGS:

None known.

SEE ALSO:

3.11 /DIO/STREAM/MODE SOREF

	name	description	type	default/range
required	SOREF	dioStream object SORef	character string	no default

Get the I/O MODE of the dioStream SOREF.

DESCRIPTION:

MODE is a readonly attribute which determines whether dioStream object SOREF reads from, or writes to its associated data stream.
Readonly attributes cannot be changed from the user interface.

The valid values of MODE are:

READONLY - Read Only
WRITEONLY - Write Only

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the dioStream interface.

RETURN:

The current value of MODE is pushed onto the STAF_RESULT stack (see SOC) and a message is printed to stdout.

EXAMPLES:

EG1. Show the current value of the MODE attribute of dioStream "DST".

staf++ dio/stream/mode DST
DIO: Stream mode = (WRITEONLY)

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the dioStream interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

3.12 /DIO/STREAM/OPEN SOREF [MODE]

	name	description	type	default/range
required	SOREF	dioStream object SORef	character string	no default
optional	MODE	Read/write mode	character string	D='R' R='R,W'

Initiate communication with a data stream. I.e. set the state to OPENED.

DESCRIPTION:

OPEN is a member function of objects which implement the dioStream interface, including:

dioFileStream - See DIO/FILESTREAM
- A data stream object associated with a disk file.
dioSockStream - See DIO/SOCKSTREAM
- A data stream object associated with a TCP/IP socket.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the dioStream interface.

MODE - Read/write mode
- Options are READING and WRITING

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the dioStream::OPEN method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Change the state of the "DST" datastream to OPENED. Set the mode to Write-only.

staf++ dio/stream/open DST W

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the dioStream interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

3.13 /DIO/STREAM/PUTEVENT SOREF [DATASET]

	name	description	type	default/range
required	SOREF	dioStream object SORef	character string	no default
optional	DATASET	In memory dataset name	character string	D='.'

Write a dataset from memory to an XDF data stream.

DESCRIPTION:

PUTEVENT is a member function of objects which implement the dioStream interface, including:

dioFileStream - See DIO/FILESTREAM
- A data stream object associated with a disk file.
dioSockStream - See DIO/SOCKSTREAM
- A data stream object associated with a TCP/IP socket.
The stream must be open and in WRITE mode.
Dataset contents (typically tables and other datasets) are written from memory to an output stream.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the dioStream interface.

DATASET - Path to an in-memory dataset name.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
dioStream::PUTEVENT
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Write the dataset Tracks contained in the dataset ProducedData to the filestream DST.

staf++ dio/stream/putevent DST ProducedData/Tracks

EG2. Write the entire ProducedData dataset to the filestream DST.

staf++ dio/stream/putevent DST ProducedData

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the dioStream interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)
BAD_MODE_OR_STATE - Attempt to write to a readonly stream or a closed stream.

BUGS:

None known.

SEE ALSO:

3.14 /DIO/STREAM/STATE SOREF

	name	description	type	default/range
required	SOREF	dioStream object SORef	character string	no default

Get the current STATE of the dioStream SOREF.

DESCRIPTION:

STATE is a readonly attribute which reflects the current state of the dioStream object SOREF.
Readonly attributes cannot be changed from the user interface.

NB. Readonly attributes are not necessarily static attributes.

Valid values of STATE are:
OPENED
CLOSED
READING
WRITING

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the dioStream interface.

RETURN:

The current value of STATE is pushed onto the STAF_RESULT stack
(see SOC) and a message is printed to stdout.

EXAMPLES:

EG1. Show the current value of the STATE attribute of
dioStream "DST".

staf++ dio/stream/state DST
DIO: Stream state = (OPENED)

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the dioStream interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

---

4. DUI, Dataset UNIX-like Interface

4.1 /DUI/APPEND SOURCE TARGET

	name	description	type	default/range
required	SOURCE	Source table name	character string	no default
required	TARGET	Target table/dataset name	character string	no default

Appends SOURCE to the end of TARGET.

DESCRIPTION:

Appends SOURCE to the end of TARGET.
TARGET must already exist.

ARGUMENTS:

SOURCE - Source table name. The name of an existing table.

TARGET - Appendee.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
duiFactory::APPEND
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Append harry onto the end of bob.

StAF DUI/APPEND harry bob

EXCEPTIONS:

SRC_NOT_FOUND - The source table doesn't exist.
TGT_DOES_NOT_EXIST - The target table doesn't exist.

BUGS:

None, it was written by Herb.

4.2 /DUI/CD [PATH]

	name	description	type	default/range
optional	PATH	Unix-like dataset path	character string	D='/dui'

CD moves through the dataset heirarchy to change the "current dataset"

DESCRIPTION:

CD is a member function of the duiFactory interface. It is
used to change the current working dataset.

ARGUMENTS:

PATH - Unix-like dataset path, either absolute, starting from the
/dui root or relative, starting from the current working dataset.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
duiFactory::CD
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. To return to the root (/dui) directory

StAF DUI/CD

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

4.3 /DUI/CP SOURCE TARGET

	name	description	type	default/range
required	SOURCE	Source table name	character string	no default

required	TARGET	Target table/dataset name	character string	no default

Make a copy of a table - put it in a specified dataset.

DESCRIPTION:

CP is a member function of the duiFactory interface which creates
a copy of an existing table with a new name. Can be in the same
or another dataset. The content of the new table will be the same
as the source.

ARGUMENTS:

SOURCE - Source table name. The name of an existing table.

TARGET - Target table/dataset name. The name of a new table or
of a dataset where the table will be placed. If a dataset is given
without a table name the new table will have the same name as the source.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
duiFactory::CP
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Copy table tsspar into dataset bob.

StAF DUI/CP tsspar bob

EXCEPTIONS:

SRC_NOT_FOUND - The source table doesn't exist.
TGT_ALREADY_EXISTS - cannot copy a table over an existing table.

BUGS:

None known.

4.4 /DUI/DF [MARKER_STRING]

	name	description	type	default/range
optional	MARKER_STRING	Any text string to mark the command in a KUMAC	character string	D='xxx'

Print the memory usage of the tables (and all other dynamically
allocated memory). The optional parameter allows tracing when
many DUI/DF commands are placed in a kumac file.

DESCRIPTION:

DF is a member function of the duiFactory interface which prints
the total memory in use.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
duiFactory::DF
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Use the command.

staf++ dui/df
92,157,884 Bytes of memory allocated
staf++ dui/df positionNumber32
92,157,884 Bytes of memory allocated (positionNumber32)

EXCEPTIONS:

BUGS:

None known.

4.5 /DUI/DU

This command is useful for finding memory-hog tables.
It lists all the tables and directories. For the tables, the
amount of allocated memory in bytes is shown (ie, maxrow x row_size).

DESCRIPTION:

DU is a member function of the duiFactory interface.

ARGUMENTS:
NONE

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
duiFactory::DU
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Dump the table space in use. Do you see the hog?

staf dui/du
/dui/BEGIN_RUN/TimeStamp ———————— 28 bytes 1 rows
/dui/BEGIN_RUN/BeginRunInfo 256 bytes 1 rows
/dui/BEGIN_RUN/SCReadout 864 bytes 6 rows
etc., etc., etc.
/dui/ProducedData/Pixels/adcxyz – 48,000,000 bytes 1000000 rows
etc., etc., etc.
Total bytes 51,610,073

BUGS:

None known.

4.6 /DUI/LN SOURCE TARGET

	name	description	type	default/range
required	SOURCE	Source table name	character string	no default
required	TARGET	Target table/dataset name	character string	no default

More guidance needed here.

DESCRIPTION:

LN is a member function of the duiFactory interface.

More guidance needed here.

ARGUMENTS:

SOURCE - Source table name.
- More guidance needed here.

TARGET - Target table/dataset name.
- More guidance needed here.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
duiFactory::LN
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. More guidance needed here.

StAF DUI/LN

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

4.7 /DUI/LS [PATH]

	name	description	type	default/range
optional	PATH	Unix-like dataset path	character string	D='.'

List the contents of the specified dataset.

DESCRIPTION:

LS is a member function of the duiFactory interface. Datasets
have the name and number of members listed. Tables have maxrowcount,
rowcount and rowsize in bytes.

ARGUMENTS:

PATH - Unix-like dataset path, absolute (from /dui) or relative.
If left blank the current working dataset is listed.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
duiFactory::LS method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. List current working dataset.

staf++ ls
DUI: Listing = ...
Name * Type * Used * Alloc'd * Size
D Switches * * 4 * -1 * -1
D Maps * * 2 * -1 * -1
D PedestalsGains * * 3 * -1 * -1
.... etc., etc., etc.

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

4.8 /DUI/MKDIR PATH

	name	description	type	default/range
required	PATH	Unix-like dataset path	character string	no default

Make a new empty dataset at the specified path.

DESCRIPTION:

MKDIR is a member function of the duiFactory interface. It creates a dataset object. The working dataset is not changed.

ARGUMENTS:

PATH - Unix-like dataset path. Absolute or relative. See DUI/CD.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
duiFactory::MKDIR
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Make a new dataset named "bob" in the current working dataset.

StAF DUI/MKDIR bob

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

4.9 /DUI/MV SOURCE TARGET

	name	description	type	default/range
required	SOURCE	Source table name	character string	no default
required	TARGET	Target dataset name	character string	no default

Move a table to a different dataset.

DESCRIPTION:

MV is a member function of the duiFactory interface. It changes the directory (dataset) but not the name of a table.

ARGUMENTS:

SOURCE - Source table name. The name of the table to move.

TARGET - Target dataset name. The new dataset to contain the table.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
duiFactory::MV
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Move the fmtpar table to dataset bob.

staf++ mv fmtpar bob

EXCEPTIONS:
SECOND_PARAM_MUST_BE_DIR - attempt to change the name of the table fails.

BUGS:

None known.

SEE ALSO:

4.10 /DUI/PRECIOUS

Marks all existing tables as precious.
See the related command RM_NONPRECIOUS.

4.11 /DUI/PWD

Print the name of the current working Directory (Dataset).

DESCRIPTION:

PWD is a member function of the duiFactory interface. It prints the
name of the current working dataset.

ARGUMENTS:

None.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
duiFactory::PWD
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Show the current dataset.

staf++ dui/pwd
DUI: Current Working Directory = (/dui/Switches)

EXCEPTIONS:

BUGS:

None known.

4.12 /DUI/RM PATH

	name	description	type	default/range
required	PATH	Unix-like table path	character string	no default

Delete a table. DUI/RM will also delete a Dataset.

DESCRIPTION:

RM is a member function of the duiFactory interface. It removes (deletes) the table or Dataset named.

ARGUMENTS:

PATH - Unix-like table path. A relative or absolute path to an existing Table or Dataset.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
duiFactory::RM
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Delete the tclpar table from dataset bob.

StAF dui/rm bob/tclpar

EXCEPTIONS:

REMOVAL_FAILED - generally if the object named does not exist.

BUGS:

I'm not sure it's supposed to delete datasets. Probably RM is a legal KUIP abbreviation for RMDIR.

4.13 /DUI/RM_NONPRECIOUS

Deletes all non-precious tables.
See the related command PRECIOUS.
Typically, you would run the PRECIOUS command before an event loop, and then run RM_NONPRECIOUS at the bottom of the loop to remove trash, ie:
DUI/PRECIOUS
top_of_loop
contents of loop
DUI/RM_NONPRECIOUS
bottom_of_loop

4.14 /DUI/RMDIR PATH

	name	description	type	default/range
required	PATH	Unix-like dataset path	character string	no default

Remove the named dataset and all its tables.

DESCRIPTION:

RMDIR is a member function of the duiFactory interface. It deletes a dataset and any tables contained therein.

ARGUMENTS:

PATH - Unix-like dataset path. A relative or absolute path to an existing Dataset.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the duiFactory::RMDIR method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Delete the "bob" dataset.

StAF DUI/RMDIR bob

EXCEPTIONS:

DIR_NOT_FOUND - the named dataset was not found.
USE_RM_FOR_TABLES_NOT_RMDIR - attempt to remove a table with dui/rmdir
BUGS:

None known.

---

5. SOC, STAF Object Catalog

5.1 /SOC/BIND PKG [SOLIB]

	name	description	type	default/range
required	PKG	Dynamically loadable package (ASP/PAM) name	character string	no default
optional	SOLIB	Sharable library name	character string	D='-'

Dynamically bind all resources for a ASP or PAM package.

DESCRIPTION:

BIND is a member function of the socCatalog interface.

On machine architectures supporting dynamic loading, the BIND function dynamically binds a shareable image for a StAF software component package.

Both ASPs and PAMs can be dynamically bound.

If PKG is an ASP, SOC/BIND will initialize the ASP (by calling ASP_init()), and then start the ASP (by calling ASP_start()). At completion of the BIND command, the ASP object factory and user interface commands are available as though the ASP had been statically linked with the STAF executable.

If PKG is a PAM, SOC/BIND will initialize the PAM (by calling PAM_init()), and then start the PAM (by calling PAM_start()). At completion of the BIND command, all PAM calculation objects and table types are available as though the PAM had been statically linked with the STAF executable.

ARGUMENTS:

PKG - Dynamically loadable package (ASP/PAM) name
- A Three Letter Acronym (TLA) denoting an Analysis Service Package
(ASP) or Physics Analysis Module (PAM) which is not currently extant
in the StAF process (see SOC/LIST).

SOLIB - Sharable library name
- A reference to the shared object file on disk which contains
the sharable resources for the PKG.
- DEFAULT: The default behavior is to search for a shared library
with the name 'libPKG.so' in the user's LD_LIBRARY_PATH.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
socCatalog::BIND
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Bind to the PAM tfs in the default location.

StAF AMI/LIST

```
+————————————————————————————————————-
+——-+—————————+—————————+——————————
— IDREF — NAME:OBJECT — TYPE:CLASS — DESCRIPTION
+——-+—————————+—————————+——————————
— 7 — pamc — amiInvoker — 2 arg.s
— 8 — pamcc — amiInvoker — 2 arg.s
— 9 — pamf — amiInvoker — 2 arg.s
+——-+—————————+—————————+——————————
```

STAF SOC/BIND tfs
tfs_filt module loaded
tfs_g2t module loaded
STAF AMI/LIST

```
+————————————————————————————————————-
+——-+—————————+—————————+——————————
— IDREF — NAME:OBJECT — TYPE:CLASS — DESCRIPTION
+——-+—————————+—————————+——————————
— 7 — pamc — amiInvoker — 2 arg.s
— 8 — pamcc — amiInvoker — 2 arg.s
— 9 — pamf — amiInvoker — 2 arg.s
— 16 — tfs_filt — amiInvoker — 1 arg.s
— 17 — tfs_g2t — amiInvoker — 8 arg.s
+——-+—————————+—————————+——————————
```

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

5.2 /SOC/COUNT

Show the current count of all registered objects.

DESCRIPTION:

COUNT is a readonly long attribute which reflects the number of all objects currently registered with the SOC object catalog.
Constructing a new object increments COUNT by 1.
Destroying an existing SOC worker object decrements COUNT by 1.

ARGUMENTS:

None.

RETURN:

The current value of COUNT is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current count of all registered objects.

StAF SOC/COUNT
SOC: Object count = 18

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

5.3 /SOC/DELETEID OID

	name	description	type	default/range
required	OID	Object ID	integer	no default

Obsolete command.

DESCRIPTION:
This command is obsolete. Please use SOC/DELETEOID instead.

SEE ALSO:
SOC/DELETEOID

5.4 /SOC/DELETEOBJECT NAME [TYPE]

	name	description	type	default/range
required	NAME	Registered object name	character string	no default
optional	TYPE	Registered interface name	character string	D='-'

Obsolete Command.

DESCRIPTION:
This command is obsolete. Please use SOC/OBJECT/DELETE instead.

DELETEOBJECT is a member function of the socCatalog interface.

DELETEOBJECT will locate a registered object with the specified NAME and TYPE and invoke that object's Destructor method.

ARGUMENTS:

NAME - Registered object name.
- More guidance needed here.

TYPE - Registered interface name.
- DEFAULT: The default behavior is to delete any (the first) object located with the specifiec NAME.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
socCatalog::DELETEOBJECT
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Delete object of TYPE == grid and NAME == chess.

StAF SOC/DELETEOBJECT chess grid

EXCEPTIONS:

BUGS:

Objects must have a LOCK attribute == FALSE to be deleted.

SEE ALSO:
SOC/OBJECT/LOCK

5.5 /SOC/DELETEOID OID

	name	description	type	default/range
required	OID	Object ID	integer	no default

Delete a registered object by OID.

DESCRIPTION:

DELETEOID is a member function of the socCatalog interface.

DELETEOBJECT will locate a registered object with the specified OID and invoke that object's Destructor method.

ARGUMENTS:

OID - Object ID
Each object registered with SOC has a unique integer ID which is listed by the soc/list command or returned by the SOC/IDOBJECT command.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
socCatalog::DELETEOID
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Delete object with OID == 123.

StAF SOC/DELETEOID 123

EXCEPTIONS:

BUGS:

Objects must have a LOCK attribute == FALSE to be deleted.

SEE ALSO:
SOC/OBJECT/LOCK

5.6 /SOC/IDOBJECT NAME [TYPE]

	name	description	type	default/range
required	NAME	Registered object name	character string	no default
optional	TYPE	Registered interface name	character string	D='-'

Identify a registered object.

DESCRIPTION:

IDOBJECT is a member function of the socCatalog interface.

IDOBJECT will return the OID of a registered object specified by SOREF
== NAME:TYPE.

ARGUMENTS:

NAME - Registered object name.

TYPE - Registered interface name.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
socCatalog::IDOBJECT
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Identify an object of TYPE dioFilestream and NAME DST.

staf++ soc/idobject DST dioFileStream
SOC: Object idRef = 101

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:
SOC/OBJECT/OID

5.7 /SOC/LIST

List all currently registered SOC worker objects.

DESCRIPTION:

Show a one-line description for each SOC worker object currently
registered with the SOC object factory in a table.

The one-line description for each object is the result of an invocation

of that object's listing method. The typical content of this listing is:
0 OID
The object's OID attribute (see SOC) presented as "%5d".
1 Lock State
The object's LOCK attribute (see SOC) presented as the
divider character between the OID column and the NAME:OBJECT
column. An object whose LOCK attribute is TRUE (cannot be
deleted) uses the "-" character, whereas an object whose
LOCK attribute is FALSE (can be deleted) uses "—" character.
2 NAME:OBJECT
The object's NAME attribute (see SOC) presented as "%-15s".
Object names longer than 15 characters are abbreviated with a
" " character at midpoint.
An object name is synonymous with an object instance.
3 TYPE:CLASS
The object's TYPE attribute (see SOC) presented as "%-15s".
Object types longer than 15 characters are abbreviated with a
" " character at midpoint.
An object type is synonymous with an object class.
4 DESCRIPTION
A class-specific description of the object. For example, for
a table, the number of rows allocated/used and the size of a
row. For a dataset the number of entries. For a filestream,
the mode, state and associated filename, etc.

Unlike the LIST command of other object factories, the SOC/LIST command
lists all registered objects whether directly instantiated by the
socCatalog object or not.

Also, unlike the LIST command of other object factories, the SOC/LIST
command lists the socCatalog (ie. the object factory) itself. Not, just
the worker objects.

ARGUMENTS:

None.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
socCatalog::list()
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. List all registered objects.

StAF SOC/LIST
+————————————————————————-
+——-+—————-+—————-+———————
— IDREF — NAME:OBJECT — TYPE:CLASS — DESCRIPTION
+——-+—————-+—————-+———————
— 0 - soc — socCatalog — 17/2048 obj.s
— 1 - spx — spxFactory — 1/2048 obj.s
— 3 - dui — duiFactory — 1/2048 obj.s
— 4 — /dui — tdmDataset — 0 ent.s
— 5 - dio — dioFactory — 0/2048 obj.s
— 6 - ami — amiBroker — 5/2048 obj.s
— 7 — pamc — amiInvoker — 2 arg.s

— 8 — pamcc — amiInvoker — 2 arg.s
— 9 — pamf — amiInvoker — 2 arg.s
— 10 — tbr — tbrFactory — 0/2048 obj.s
— 11 — tbr_MotifViewer — tbrMotifViewer —
— 12 - tnt — tntFactory — 0/2048 obj.s
— 13 - top — topFactory — 0/2048 obj.s
— 14 — bob — socObject —
— 15 — chess — spxGrid — Size = (16, 16)
— 16 — tfs_filt — amiInvoker — 1 arg.s
— 17 — tfs_g2t — amiInvoker — 8 arg.s
+——-+—————+—————+———————

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

5.8 /SOC/NEWOBJECT NAME

	name	description	type	default/range
required	NAME	Name for new socObject object	character string	no default

Create a new socObject object.

DESCRIPTION:

Each socObject created by the socCatalog shows up as an object
managed by the socCatalog (see SOC/COUNT and SOC/LIST) and
registered with the socCatalog (see SOC/COUNT and SOC/LIST).

ARGUMENTS:

NAME - Case-sensitive alphanumeric name for new socObject object.
- Use this name as part of SOREF (see SOC) to specify this particular
socObject object in subsequent commands.
- More guidance needed here, most definitely, as to just what an
undifferentiated socObject is good for.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
socCatalog::newObject
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Create a new socObject with NAME "bob"

StAF SOC/NEWOBJECT bob

EXCEPTIONS:

OBJECT_NOT_CREATED - The object creation failed. See error stack for
detailed explanation of failure.

BUGS:

None known.

SEE ALSO:

SOC/OBJECT

5.9 /SOC/OBJECT/DELETE NAME TYPE

	name	description	type	default/range
required	NAME	Registered Object Name.	character string	no default
required	TYPE	Known Object Type.	character string	no default

Directly invoke the destructor method of object NAME:TYPE.

DESCRIPTION:

DELETE is a member function of objects which implement the socObject
interface.

More guidance needed here.

ARGUMENTS:

NAME - Registered Object Name.

TYPE - Known Object Type.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
socObject::DELETE
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Delete registered object "chess:spxGrid".

StAF SOC/OBJECT/DELETE chess spxGrid

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by NAME:TYPE can be found which
implements the socObject interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

Fails to delete stream objects?
staf++ soc/object/delete bob dioFileStream
*** Break *** Segmentation violation
Interrupt trace routine not available
*** Break *** Simulated break
SOC/OBJECT/DELETE ought to use SOREF instead of NAME & TYPE.
Objects must have a LOCK attribute == FALSE to be deleted.

SEE ALSO:
SOC/DELETEOBJECT
SOC/DELETEOID

5.10 /SOC/OBJECT/IMPLEMENTS OID INTERFACE

	name	description	type	default/range
required	OID	socObject object ID	integer	no default
required	INTERFACE	Known interface name	character string	no default

Inquire whether object #OID implements interface IFACENAME.

DESCRIPTION:

IMPLEMENTS is a member function of all registered component objects in StAF.

If object #OID's interface is a subclass of another interface, then object #OID implements that base class interface as well as it's own. This means that object #OID can be treated as an object of the base class when appropriate.

E.G. A dioFileStream object is a subclass of a dioStream object. Hence, any file stream object can be treated as a generic stream object. I.E. An object created with the DIO/NEWFILESTREAM command can be manipulated with the DIO/STREAM/* commands as well as the DIO/FILESTREAM/* commands.

ARGUMENTS:

OID - Object ID (see SOC).
- denoting an object implementing the socObject interface.

INTERFACE - Known interface name.
- The name of an interface (eg. name of TYPE) known to StAF.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
socObject::IMPLEMENTS
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Inquire whether object #14 implements the socObject interface.

StAF SOC/OBJECT/IMPLEMENTS 14 socObject
SOC: Object (bob) DOES implement (socObject)

EG2. Inquire whether object #14 implements the socCatalog interface.

StAF SOC/OBJECT/IMPLEMENTS 14 socCatalog
SOC: Object (bob) DOES NOT implement (socCatalog)

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by #OID can be found which implements the socObject interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

SOC/OBJECT/IMPLEMENTS ought to use SOREF instead of OID.
- The value of OID can be determined by using SOC/OBJECT/OID

SEE ALSO:

5.11 /SOC/OBJECT/LOCK OID [NEW_VALUE]

	name	description	type	default/range
required	OID	socObject object ID	integer	no default
optional	NEW_VALUE	New value of LOCK attribute	character string	D='-' R='-,T,F'

Get or set the LOCK attribute of the socObject SOREF.

DESCRIPTION:

LOCK is a read-writable attribute which determines whether an object can be deleted from the system. Under normal behavior, attempting to delete a locked object (LOCK == TRUE) will fail.

To get the current value of LOCK, leaving LOCK unchanged, do not specify a new value in the optional argument NEW_VALUE.

To set a new value of LOCK, specify the new value as the optional argument NEW_VALUE.

ARGUMENTS:

OID - Object ID (see SOC).
- denoting an object implementing the socObject interface.

NEW_VALUE - New value for the LOCK attribute.
- LOCK == T(rue) denotes the object cannot be deleted.
- LOCK == F(alse) denotes the object can be deleted.
- DEFAULT: Show the current value of LOCK, do not change it.

RETURN:

The current value of LOCK is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current value of the LOCK attribute of socObject object with OID == 14.

StAF SOC/OBJECT/LOCK 14
SOC: Object lock = FALSE

EG2. Lock socObject object with OID == 14.

StAF SOC/OBJECT/LOCK 14 T

EG3. Unlock socObject object with OID == 14.

StAF SOC/OBJECT/LOCK 14 F

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by #OID can be found which implements the socObject interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

SOC/OBJECT/LOCK ought to use SOREF instead of OID. Doesn't actually return OBJECT_NOT_FOUND but KAM_INVALID_IDREF.

SEE ALSO:

5.12 /SOC/OBJECT/NAME OID

	name	description	type	default/range
required	OID	socObject object ID	integer	no default

Get the NAME attribute of the socObject #OID.

DESCRIPTION:

NAME is a readonly attribute which reflects the value of the NAME attribute of the socObject #OID. Readonly attributes cannot be changed from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

OID - Object ID (see SOC).
- denoting an object implementing the socObject interface.

RETURN:

The current value of NAME is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current value of the NAME attribute of socObject 99.

staf++ soc/object/name 99
SOC: Object name = /dui/BEGIN_RUN/GN6

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by #OID can be found which implements the socObject interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

SOC/OBJECT/NAME ought to use SOREF instead of OID.
- The value of OID can be determined by using SOC/OBJECT/OID.
Doesn't actually return OBJECT_NOT_FOUND but KAM_INVALID_IDREF.

SEE ALSO:

5.13 /SOC/OBJECT/OID NAME [TYPE]

	name	description	type	default/range
required	NAME	Registered Object Name.	character string	no default
optional	TYPE	Known Object Type.	character string	D='-'

Get the OID attribute of the socObject NAME:TYPE.

DESCRIPTION:

OID is a readonly attribute which reflects the value of the OID attribute of the socObject NAME:TYPE. Readonly attributes cannot be changed from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

NAME - Registered Object Name.

TYPE - Known Object Type.

RETURN:

The current value of OID is pushed onto the STAF_RESULT stack
(see SOC).

EXAMPLES:

EG1. Show the current value of the OID attribute of object
"chess:spxGrid".

StAF SOC/OBJECT/OID chess spxGrid
SOC: Object OID = 14

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by NAME:TYPE can be found which
implements the socObject interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

SOC/OBJECT/OID ought to use SOREF instead of NAME & TYPE.

SEE ALSO:

5.14 /SOC/OBJECT/TYPE OID

	name	description	type	default/range
required	OID	socObject object ID	integer	no default

Get the TYPE attribute of the socObject #OID.

DESCRIPTION:

TYPE is a readonly attribute which reflects the value of the TYPE
attribute of the socObject #OID. Readonly attributes cannot be changed
from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

OID - Object ID (see SOC).
- denoting an object implementing the socObject interface.

RETURN:

The current value of TYPE is pushed onto the STAF_RESULT stack
(see SOC).

EXAMPLES:

EG1. Show the current value of the TYPE attribute of the object with OID 99;

staf++ soc/object/type 99
SOC: Object type = tdmTable

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by #OID can be found which

implements the socObject interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

SOC/OBJECT/TYPE ought to use SOREF instead of OID.
- The value of OID can be determined by using SOC/OBJECT/OID.

SEE ALSO:

5.15 /SOC/OBJECT/VERSION OID

	name	description	type	default/range
required	OID	socObject object ID	integer	no default

Get the VERSION attribute of the socObject #OID.

DESCRIPTION:

VERSION is a readonly attribute which reflects the value of the VERSION
attribute of the socObject #OID. Readonly attributes cannot be changed
from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

OID - Object ID (see SOC).
- denoting an object implementing the socObject interface.

RETURN:

The current value of VERSION is pushed onto the STAF_RESULT stack
(see SOC).

EXAMPLES:

EG1. Show the current value of the VERSION attribute of
socObject OID == 14.

StAF SOC/OBJECT/VERSION 14
SOC: Object version = dev

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by #OID can be found which
implements the socObject interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

SOC/OBJECT/VERSION ought to use SOREF instead of OID.
- The value of OID can be determined by using SOC/OBJECT/OID.

SEE ALSO:

5.16 /SOC/RELEASE PKG

	name	description	type	default/range
required	PKG	Dynamically loaded package (ASP/PAM) name	character string	no default

Release a dynamically bound ASP or PAM package.

DESCRIPTION:

RELEASE is a member function of the socCatalog interface.

More guidance needed here.

ARGUMENTS:

PKG - Dynamically loaded package (ASP/PAM) name.
- More guidance needed here.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
socCatalog::RELEASE
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Release the tfs PAM.

StAF SOC/RELEASE tfs

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:
SOC/BIND

---

6. SPX, Service Package eXample

6.1 /SPX/COUNT

Show count of known SPX objects.

6.2 /SPX/DUMMY/NCALLS DNAME

	name	description	type	default/range
required	DNAME	spxDummy object name.	character string	no default

Shows number of calls to spxDummy functions.

6.3 /SPX/DUMMY/NULL DNAME

	name	description	type	default/range

required	DNAME	spxDummy object name.	character string	no default

Does nothing.

6.4 /SPX/GRID/GET DNAME M N

	name	description	type	default/range
required	DNAME	spxGrid object name.	character string	no default
required	M	First index of cell.	integer	no default
required	N	Second index of cell.	integer	no default

Get cell value of spxGrid object.

6.5 /SPX/GRID/HEIGHT DNAME

	name	description	type	default/range
required	DNAME	spxGrid object name.	character string	no default

Show height attribute of spxGrid object.

6.6 /SPX/GRID/SET DNAME M N VALUE

	name	description	type	default/range
required	DNAME	spxGrid object name.	character string	no default
required	M	First index of cell.	integer	no default
required	N	Second index of cell.	integer	no default
required	VALUE	New value of cell.	integer	no default

Set cell value of spxGrid object.

6.7 /SPX/GRID/WIDTH DNAME

	name	description	type	default/range
required	DNAME	spxGrid object name.	character string	no default

Show width attribute of spxGrid object.

6.8 /SPX/LIST

List all known SPX objects.

6.9 /SPX/NEWDUMMY NAME

	name	description	type	default/range
required	NAME	spxDummy name.	character string	no default

Create a new spxDummy object.

6.10 /SPX/NEWGRID NAME HEIGHT WIDTH

	name	description	type	default/range
required	NAME	spxGrid name.	character string	no default
required	HEIGHT	Grid height.	integer	no default
required	WIDTH	Grid width.	integer	no default

Create a new spxGrid object.

---

7. TDM, Table and Dataset Manager

7.1 /TDM/ALLOCSTATS

Print statistics of dataset and table usage.

DESCRIPTION:

ALLOCSTATS is a member function of the tdmFactory interface.
Numbers printed are maybe of some value to debuggers (people, not DBX).

ARGUMENTS:

None.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
tdmFactory::ALLOCSTATS
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1.
StAF TDM/ALLOCSTATS
AllocStats: bufSize 0, dsetSize 51606632, listSize 0, memCalls 826, tidSize 85199

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

7.2 /TDM/COUNT

Show the current count of TDM worker objects.

DESCRIPTION:

COUNT is a readonly long attribute which reflects the number of TDM worker objects currently registered with the TDM object factory. Constructing a new TDM worker object increments COUNT by 1. Destroying an existing TDM worker object decrements COUNT by 1.

TDM worker objects include:
tdmDataset - See TDM/DATASET
- C++ representation of DSL datasets.
tdmTable - See TDM/TABLE
- C++ representation of DSL tables.

ARGUMENTS:

None.

RETURN:

The current value of COUNT is pushed onto the STAF_RESULT stack (see SOC). A message is also printed to stdout.

EXAMPLES:

EG1. Show the current count of TDM worker objects.

StAF TDM/COUNT
TDM: Object count = 18

BUGS:
None known.

7.3 /TDM/DATASET/ADDDATASET SOREF NAME

	name	description	type	default/range
required	SOREF	tdmDataset object SORef	character string	no default
required	NAME	Name for new tdmDataset object	character string	no default

Not Yet Implemented. Intended to copy a dataset?

DESCRIPTION:

ADDDATASET is a member function of objects which implement the tdmDataset interface.

More guidance needed here.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the tdmDataset interface.

NAME - Name for new tdmDataset object

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
tdmDataset::ADDDATASET
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Invoke the ADDDATASET method function of tdmDataset "bob"

More guidance needed here.

StAF TDM/DATASET/ADDDATASET bob

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the tdmDataset interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

7.4 /TDM/DATASET/ADDTABLE SOREF NAME SPEC MAXROWCOUNT

	name	description	type	default/range
required	SOREF	tdmDataset object SORef	character string	no default
required	NAME	Name for new tdmTable object	character string	no default
required	SPEC	Type specifier for a table type	character string	no default
required	MAXROWCOUNT	Count of rows allocated in memory	integer	R='0:'

Not implemented yet. Intended to add tables to datasets? See TDM/NEWTABLE

DESCRIPTION:

ADDTABLE is a member function of objects which implement the tdmDataset interface.

More guidance needed here.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the tdmDataset interface.

NAME - Name for new tdmTable object
SPEC - Type specifier for a table type.

MAXROWCOUNT - Count of rows allocated in memory.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the tdmDataset::ADDTABLE
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Invoke the ADDTABLE method function of tdmDataset "Geometry"

tdm/dataset/addtable Geometry tpg_detector tpg_detector 5
NOT_YET_IMPLEMENTED-/afs/rhic/.../asps/staf/tdm/src/tdmClasses.c

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the tdmDataset interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

7.5 /TDM/DATASET/ENTRYCOUNT SOREF

	name	description	type	default/range
required	SOREF	tdmDataset object SORef	character string	no default

Get the ENTRYCOUNT of a tdmDataset.

DESCRIPTION:

The ENTRYCOUNT is the total number of tables and other datasets contained in the dataset.
ENTRYCOUNT is a readonly attribute which reflects the value of the ENTRYCOUNT attribute of the tdmDataset SOREF.
Readonly attributes cannot be changed from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the tdmDataset interface.

RETURN:

The current value of ENTRYCOUNT is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current value of the ENTRYCOUNT attribute of tdmDataset "Maps".

staf++ tdm/dataset/entrycount Maps
TDMDATASET: Entry Count = 2

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the tdmDataset interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

7.6 /TDM/DATASET/NAME SOREF

	name	description	type	default/range
required	SOREF	tdmDataset object SORef	character string	no default

Get the NAME attribute of the tdmDataset SOREF. (Redundant?)

DESCRIPTION:

NAME is a readonly attribute which reflects the value of the NAME
attribute of the tdmDataset SOREF. Readonly attributes cannot be changed
from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the tdmDataset interface.

RETURN:

The current value of NAME is pushed onto the STAF_RESULT stack
(see SOC).

EXAMPLES:

EG1.
Staf tdm/dataset/name ProducedData
TDMDATASET: DSL name = (ProducedData)

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the tdmDataset interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

7.7 /TDM/DATASET/SHOW SOREF

	name	description	type	default/range
required	SOREF	tdmDataset object SORef	character string	no default

Not implemented yet. No idea what its for.

DESCRIPTION:

SHOW is a member function of objects which implement the tdmDataset
interface.

More guidance needed here.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the tdmDataset interface.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
tdmDataset::SHOW
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Invoke the SHOW method function of tdmDataset "bob"
More guidance needed here.

StAF TDM/DATASET/SHOW bob

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the tdmDataset interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

7.8 /TDM/LIST

List all currently registered TDM worker objects.

DESCRIPTION:

Show a tabular one-line description for each TDM worker object currently registered with the TDM object factory. Note the DUI/cd, ls and pwd commands give a UNIX-like access to the lists of Datasets and Tables.

The one-line description for each object is the result of an invokation of that object's listing method. The typical content of this listing is:
0 OID
The object's OID attribute (see SOC) presented as "%5d".
1 Lock State
The object's LOCK attribute (see SOC) presented as the divider character between the OID column and the NAME:OBJECT column. An object whose LOCK attribute is TRUE (cannot be deleted) uses the "-" character, whereas an object whose LOCK attribute is FALSE (can be deleted) uses "—" character.
2 NAME:OBJECT
The object's NAME attribute (see SOC) presented as "%-15s".
Object names longer than 15 characters are abreviated with a " " character at midpoint.
An object name is synonymous with an object instance.
3 TYPE:CLASS
The object's TYPE attribute (see SOC) presented as "%-15s".
Object types longer than 15 characters are abreviated with a " " character at midpoint.
An object type is synonymous with an object class.
4 DESCRIPTION
A class-specific description of the object. For Datasets the number of entries are shown and for tables the number of used and allocated rows and the number of bytes of each row.

TDM worker objects include:
tdmDataset - See TDM/DATASET
- C++ representation of DSL datasets.
tdmTable - See TDM/TABLE

- C++ representation of DSL tables.

ARGUMENTS:
None.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
tdmFactory::list()
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. List all currently registered TDM worker objects.

STAF[3] tdm/list
+———————————————————
+——-+—————+—————+———————-
— IDREF — NAME:OBJECT — TYPE:CLASS — DESCRIPTION
+——-+—————+—————+———————-
— 3 — /dui — tdmDataset — 0 ent.s
+——-+—————+—————+———————-
EXCEPTIONS:

BUGS:

None known.

SEE ALSO:
DUI/LS

7.9 /TDM/NEWDATASET NAME

	name	description	type	default/range
required	NAME	Name for new tdmDataset object	character string	no default

Create a new tdmDataset object.

DESCRIPTION:

Each tdmDataset created by the tdmFactory shows up as an object
managed by the tdmFactory (see TDM/COUNT and TDM/LIST) and
registered with the socCatalog (see SOC/COUNT and SOC/LIST).

ARGUMENTS:

NAME - Case-sensitive alphanumeric name for new tdmDataset object.
- Use this name as part of SOREF (see SOC) to specify this particular
tdmDataset object in subsequent commands.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
tdmFactory::newDataset method is pushed onto the
STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Create a new tdmDataset with NAME ”bob”

StAF TDM/NEWDATASET bob

EXCEPTIONS:

OBJECT_NOT_CREATED - The object creation failed. See error stack for detailed explanation of failure.

BUGS:

None known.

SEE ALSO:

TDM/DATASET

7.10 /TDM/NEWTABLE NAME SPEC MAXROWCOUNT

	name	description	type	default/range
required	NAME	Name for new tdmTable object	character string	no default
required	SPEC	Type specifier for a table type	character string	no default
required	MAXROWCOUNT	Count of rows allocated in memory	integer	R='0:'

Create a new tdmTable object in the current Dataset.

DESCRIPTION:

Each tdmTable created by the tdmFactory shows up as an object managed by the tdmFactory (see TDM/COUNT and TDM/LIST) and registered with the socCatalog (see SOC/COUNT and SOC/LIST).

ARGUMENTS:

NAME - Case-sensitive alphanumeric name for new tdmTable object. - Use this name as part of SOREF (see SOC) to specify this particular tdmTable object in subsequent commands.

SPEC - Type specifier for a table type. This can either be simply the name of an existing table type or a complete definition of a new table type.

MAXROWCOUNT - Number of table rows to be allocated in memory.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
tdmFactory::newTable
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Create a new Table in Dataset "bob" of type "tpt_spars" with 100 rows

STAF[19] dui/cd bob
STAF[20] tdm/newtable george tpt_spars 100
STAF[21] dui/ls
DUI: Listing = ...
Name * Type * Used * Alloc'd * Size
T george * tpt_spars * 0 * 100 * 216

EXCEPTIONS:

OBJECT_NOT_CREATED - The object creation failed. See error stack for detailed explanation of failure.

BUGS:

None known.

SEE ALSO:

TDM/TABLE

7.11 /TDM/TABLE/CELL/GETVALUE SOREF [SCREEN_SWITCH]

	name	description	type	default/range
required	SOREF	tdmTable.CELL component SORef	character string	no default
optional	SCREEN_SWITCH	Screen output. Either OFF_SCREEN or ON_SCREEN.	character string	D='OFF_SCREEN'

Return the value contained in a single cell of a table.
SCREEN_SWITCH controls whether the returned value is written
to the screen.

DESCRIPTION:

GETVALUE is a member function of CELL components of objects which
implement the tdmTable interface whcih returns the contents of the CELL.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting a CELL component of an object implementing the
tdmTable interface. Must be enclosed in single quotes with the
row number in square brackets and the column name preceded by a
period. If the row number is the content of a KUIP variable
(a common usage) the need to include the variable in square brackets
requires the ugly format 'table_name['//[row_variable]//'].col_name

RETURN:

The contents of the cell are returned in staf_result(1)
A message with the contents is also printed to stdout.
Success (STAFCV_OK) or failure (STAFCV_BAD) of the
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. STAF tdm/table/cell/getvalue 'tpt_spars[0].last_row' ON_SCREEN
TDMTABLE: Cell data = 45

EXCEPTIONS:

OBJECT_NOT_FOUND - No CELL component specified by SOREF can be
found for an object which implements the tdmTable interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)
INVALID_TABLE_COLUMN - Found the table but not the cell.

BUGS:

It is inconvenient that the contents of the cell are printed (good
for interactive use but for production there is no way to suppress this?)
and to get the value for use in a kumac it is necessary to fetch it from
staf_result(1).
An ill-formed cell reference causes a segmentation fault.

STAF[55] tdm/table/cell/get 'tpg_cathode.cath_mat'
*** Break *** Segmentation violation
TRACEQ. In-line trace-back still not available.

SEE ALSO:

7.12 /TDM/TABLE/CELL/PUTVALUE SOREF VALUES

	name	description	type	default/range
required	SOREF	tdmTable.CELL component SORef	character string	no default
required	VALUES	List of new cell values	character string	no default

Insert data into a cell of a table.

DESCRIPTION:

PUTVALUE can be used to change the contents of one cell of a table.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting a CELL component of a Table. Note the table row is enclosed in square brackets and the column name should follow a period. The whole SOREF is enclosed in single quotes. If the row number is the content of a KUIP variable (a common usage) the need to include the variable in square brackets requires the ugly format
'table_name['//[row_variable]//'].col_name

VALUES - List of new cell values. Unless the cell is defined as an array, this will be a single VALUE. For an array cell, all members must be listed, separated by blanks.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1.
tdm/table/cell/putvalue 'tpt_spars[0].nskip' 10

EG2.
tdm/table/cell/putvalue 'tpt_spars[0].skip' 1 2 3 4 5 6 7 8 9 10

EXCEPTIONS:

OBJECT_NOT_FOUND - Table name is wrong or SOREF not in single quotes.
INVALID_TABLE_COLUMN - CELL specified by SOREF not found.

BUGS:

Ill-formed SOREF's produce Segmentation violation errors.
If the variable is an array, unspecified members are often filled with arbitrary junk. Makes it impractical to use interactively on large arrays. It would be nice to have a more flexible input structure like 1 2 3 5*0 11*1 9 8 2 - or something like that.

SEE ALSO:

7.13 /TDM/TABLE/COLUMNCOUNT SOREF

	name	description	type	default/range
required	SOREF	tdmTable object SORef	character string	no default

Get the COLUMNCOUNT (number of variables in a row) of a Table.

DESCRIPTION:

COLUMNCOUNT is a readonly attribute which reflects the value of the COLUMNCOUNT attribute of the tdmTable SOREF. Readonly attributes cannot be changed from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the tdmTable interface.

RETURN:

The current value of COLUMNCOUNT is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current COLUMNCOUNT of Table "george".

STAF[33] tdm/table/columncount george
TDMTABLE: Column Count = 10
EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the tdmTable interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

7.14 /TDM/TABLE/DUMP SOREF NROWS IFIRST NAMEOFFILE COLUMNLIST

	name	description	type	default/range
required	SOREF	name of table	character string	no default
required	NROWS	Number of rows to dump	integer	D=10
required	IFIRST	First row to dump	integer	D=0
required	NAMEOFFILE	Name of output file	character string	no default
required	COLUMNLIST	List of columns	character string	no default

Dumps a table to file.

The IFIRST parameter counts from zero _UNLIKE_ Fortran.

If you want all the rows, use a large number for NROWS, and zero for IFIRST.

The COLUMNLIST parameter is used
to select a subset of the columns. In the COLUMNLIST parameter, separate the
column names with carets ( ). See the example below.
Instead of a list of columns, you can type allColumns.

EXAMPLE:
This example writes columns id, offset, and pedestal of
rows 0 through 9 of the table tpg_cathode to a file named
myfile.dat.

STAF[46] tdm/table/dump tpg_cathode 10 0 myfile.dat id offset pedestal

EXCEPTIONS:

OBJECT_NOT_FOUND - Table not found.

BUGS:

None known.

7.15 /TDM/TABLE/MAXROWCOUNT SOREF [NEW_VALUE]

	name	description	type	default/range
required	SOREF	tdmTable object SORef	character string	no default
optional	NEW_VALUE	New value of MAXROWCOUNT attribute	integer	D=-1 R='-1:'

Get or set the MAXROWCOUNT of a Table. I.e. the size.

DESCRIPTION:

MAXROWCOUNT is a read-writable attribute which determines the value of
the MAXROWCOUNT attribute.

To get the current value of MAXROWCOUNT, leaving MAXROWCOUNT unchanged, do
not
specify a new value in the optional argument NEW_VALUE.

To set a new value of MAXROWCOUNT, specify the new value as the optional
argument NEW_VALUE.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the tdmTable interface.

NEW_VALUE - New value for the MAXROWCOUNT attribute.
- DEFAULT: Show the current value of MAXROWCOUNT, do not change it.

RETURN:

The current value of MAXROWCOUNT is pushed onto the STAF_RESULT stack
(see SOC).

EXAMPLES:

EG1. Show the current MAXROWCOUNT of Table "george"

STAF[29] tdm/table/maxrowcount george
TDMTABLE: Max Row Count = 100

EG2. Set the MAXROWCOUNT attribute of tdmTable object "bob" to 123.

StAF TDM/TABLE/MAXROWCOUNT bob 123

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the tdmTable interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

7.16 /TDM/TABLE/NAME SOREF

	name	description	type	default/range
required	SOREF	tdmTable object SORef	character string	no default

Get the NAME attribute of the tdmTable SOREF. Usefulness?

DESCRIPTION:

NAME is a readonly attribute which reflects the value of the NAME attribute of the tdmTable SOREF. Readonly attributes cannot be changed from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the tdmTable interface.

RETURN:

The current value of NAME is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current value of the NAME attribute of tdmTable "george".

STAF[34] tdm/table/name george
TDMTABLE: DSL Name = (george)

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the tdmTable interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

7.17 /TDM/TABLE/PRINT SOREF [NROWS IFIRST]

	name	description	type	default/range
required	SOREF	tdmTable object SORef	character string	no default
optional	NROWS	Number of rows to print	integer	D=10
optional	IFIRST	First row to print	integer	D=0

Print the contents (or some sequential rows) of a table.

DESCRIPTION:

PRINT is a member function of objects which implement the tdmTable
interface which causes the object to print its contents.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the tdmTable interface.
i. e. the Table name.

NROWS - Number of rows to print.

IFIRST - First row to print.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
tdmTable::PRINT
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Invoke the PRINT method function of tdmTable "tpg_cathode"

STAF[46] tdm/table/print tpg_cathode
ROW # cath_mat cath_in_rad cath_out_rad cath_thick
0: -4 46.825 200 0.00762

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the tdmTable interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

7.18 /TDM/TABLE/ROWCOUNT SOREF [NEW_VALUE]

	name	description	type	default/range
required	SOREF	tdmTable object SORef	character string	no default
optional	NEW_VALUE	New value of ROWCOUNT attribute	integer	D=-1 R='-1:'

Get or set the ROWCOUNT of a Table. I.e. the number used.

DESCRIPTION:

ROWCOUNT is a read-writable attribute which determines the value of
the ROWCOUNT attribute.

To get the current value of ROWCOUNT, leaving ROWCOUNT unchanged, do not
specify a new value in the optional argument NEW_VALUE.

To set a new value of ROWCOUNT, specify the new value as the optional
argument NEW_VALUE. NEW_VALUE cannot be greater than MAXROWCOUNT.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).

- denoting an object implementing the tdmTable interface.

NEW_VALUE - New value for the ROWCOUNT attribute.
- DEFAULT: Show the current value of ROWCOUNT, do not change it.

RETURN:

The current value of ROWCOUNT is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current ROWCOUNT of Table "george"

STAF[32] tdm/table/rowcount george
TDMTABLE: Row Count = 0

EG2. Set the ROWCOUNT attribute of tdmTable object "bob" to 123.

StAF TDM/TABLE/ROWCOUNT bob 123

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the tdmTable interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)
INVALID_ROW_COUNT - attempt to set rowcount larger than MAXROWCOUNT.

BUGS:

None known.

7.19 /TDM/TABLE/ROWSIZE SOREF

	name	description	type	default/range
required	SOREF	tdmTable object SORef	character string	no default

Get the ROWSIZE (bytes per row) of a Table.

DESCRIPTION:

ROWSIZE is a readonly attribute which reflects the value of the ROWSIZE attribute of the tdmTable SOREF. Readonly attributes cannot be changed from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the tdmTable interface.

RETURN:

The current value of ROWSIZE is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current ROWSIZE of Table "george".

STAF[35] tdm/table/rowsize george
TDMTABLE: Row Size = 216 bytes

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the tdmTable interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

DUI/LS

7.20 /TDM/TABLE/SHOW SOREF

	name	description	type	default/range
required	SOREF	tdmTable object SORef	character string	no default

Show the type definition of a table.

DESCRIPTION:

SHOW is a member function of objects which implement the tdmTable interface which prints the definition of the table type.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the tdmTable interface.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
tdmTable::SHOW
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1.
STAF tdm/table/show tphit
TDMTABLE: Table = ...
struct tcl_tphit
long cluster, flag, id, id_globtrk, nseq, row, track;
float alpha, dalpha, dlambda, dq, dx, dy, dz, lambda, phi, prf, q, x, y, z, zrf;

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the tdmTable interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:
TDM/TABLE/SPECIFIER gives similar information.

7.21 /TDM/TABLE/SPECIFIER SOREF

	name	description	type	default/range
required	SOREF	tdmTable object SORef	character string	no default

Get and Print the type SPECIFIER of a tdmTable.

DESCRIPTION:

SPECIFIER is a readonly attribute which reflects the value of the SPECIFIER attribute of the tdmTable SOREF. Readonly attributes cannot be changed from the user interface. The specifier is the Table "type"

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the tdmTable interface.

RETURN:

The current value of SPECIFIER is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the type SPECIFIER of Table "george".

STAF[36] tdm/table/specifier george
TDMTABLE: Type Specifier = ...
struct tpt_spars
long first_row, last_row, nskip, skip[45], hole, nmin, ilimit;
float oy, oz, outlimit;

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the tdmTable interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

7.22 /TDM/TABLE/TYPENAME SOREF

	name	description	type	default/range
required	SOREF	tdmTable object SORef	character string	no default

Get the TYPENAME of a Table.

DESCRIPTION:

TYPENAME is a readonly attribute which reflects the value of the TYPENAME attribute of the tdmTable SOREF. Readonly attributes cannot be changed from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the tdmTable interface.

RETURN:

The current value of TYPENAME is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current TYPENAME of Table "george".

STAF[37] tdm/table/typename george
TDMTABLE: Type Name = (tpt_spars)

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the tdmTable interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

7.23 /TDM/TABLE/TYPESPECIFIERS/LIST [TYPEID]

	name	description	type	default/range
optional	TYPEID	Table type ID	integer	D=-1

List one or all (ID negative) table type names.

DESCRIPTION:

LIST is a member function of objects which implement the tdmTypespecifiers interface. Reminds user of which types have been defined in case he forgot the spelling of the name.

ARGUMENTS:

TYPEID - Table type ID.
Only useful value would seem to be the default (-1)

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
tdmTypespecifiers::LIST
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. List the third type in the current list.

STAF[57] tdm/type/list 3
TDM: Type name = (tpt_track)

EXCEPTIONS:

INVALID_TYPE_ID ID specified is out of range - I.e. larger than total number of Types defined.

BUGS:

None known.

7.24 /TDM/TABLE/TYPESPECIFIERS/LOAD IDL_FILE

	name	description	type	default/range
required	IDL_FILE	IDL file containing table IDL	character string	———————————-

Read Table type specifier data from an IDL file.

DESCRIPTION:

LOAD is a member function of objects which implement the tdmTypespecifiers
interface which reads a type specification from an idl format file.

ARGUMENTS:

IDL_FILE - IDL file containing table IDL.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
tdmTypespecifiers::LOAD
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

StaF tdm/type/load tpg_transform.idl
file = tpg_transform.idl
—
struct tpg_transform
float global_origin[3];
float local_origin[3];
float phi_limhi;
float phi_limlo;
float sector_angle;
float sector_cos;
float sector_sin;
float y_local_limhi;
float y_local_limlo;
float z_global_limhi;
float z_global_limlo;
float z_local_limhi;
;

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the tdmTypespecifiers interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

7.25 /TDM/TABLE/TYPESPECIFIERS/SHOW [TYPENAME]

	name	description	type	default/range
optional	TYPENAME	Table type name	character string	D='*'

Print the definition of the Table type name given.

DESCRIPTION:

SHOW is a member function of objects which implement the tdmTypespecifiers interface. Lists the data types and names of the columns of a table type.

ARGUMENTS:

TYPENAME - Table type name.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
tdmTypespecifiers::SHOW
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. .
StAF TDM/TYPESPECIFIERS/SHOW tpt_spars
TDM: Type spec = ...
struct tpt_spars
long first_row, last_row, nskip, skip[45], hole, nmin, ilimit;
float oy, oz, outlimit;

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the tdmTypespecifiers interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

TDM/TABLE/SHOW

---

8. TNT, Tables to NTuples

8.1 /TNT/COUNT

Show the current count of TNT worker objects.

DESCRIPTION:

COUNT is a readonly long attribute which reflects the number of TNT
worker objects currently registered with the TNT object factory.
Constructing a new TNT worker object increments COUNT by 1,
destroying an existing TNT worker object decrements COUNT by 1.

TNT worker objects now include:
tntCWNtuple.

ARGUMENTS:

None.

RETURN:

The current value of COUNT is pushed onto the STAF_RESULT stack
(see SOC).

EXAMPLES:

EG1. Show the current count of TNT worker objects.

StAF TNT/COUNT
TNT: Object count = 18

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

8.2 /TNT/CWNTUPLE/APPEND HID TABLE

	name	description	type	default/range
required	HID	HBOOK ID for CWNtuple.	integer	no default
required	TABLE	tdmTable name	character string	no default

Add the contents of the table to the current contents of the ntuple.

DESCRIPTION:

APPEND is a member function of objects which implement the tntCwntuple
interface. The contents of the table are added to the existing contents
of the ntuple.

ARGUMENTS:

HID - HBOOK ID for CWNtuple.
- denoting an object implementing the tntCwntuple interface.

TABLE - tdmTable name.
- A table whose type matches the definition of the ntuple.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
tntCwntuple::APPEND
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Invoke the APPEND method function of tntCwntuple 100 on table bob

StAF TNT/CWNTUPLE/APPEND 100 bob

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by HID can be found which
implements the tntCwntuple interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

8.3 /TNT/CWNTUPLE/COLUMNCOUNT HID

	name	description	type	default/range
required	HID	HBOOK ID for CWNtuple.	integer	no default

Get the COLUMNCOUNT attribute of the tntCwntuple HID.

DESCRIPTION:

COLUMNCOUNT is a readonly attribute which reflects the value of the COLUMNCOUNT attribute of the tntCwntuple HID. Readonly attributes cannot be changed from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

HID - HBOOK ID for CWNtuple.
- denoting an object implementing the tntCwntuple interface.

RETURN:

The current value of COLUMNCOUNT is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current value of the COLUMNCOUNT attribute of tntCwntuple 100.

StAF TNT/CWNTUPLE/COLUMNCOUNT 100

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by HID can be found which implements the tntCwntuple interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

8.4 /TNT/CWNTUPLE/ENTRYCOUNT HID

	name	description	type	default/range
required	HID	HBOOK ID for CWNtuple.	integer	no default

Get the ENTRYCOUNT attribute of tntCwntuple HID. I.e. number of rows filled.

DESCRIPTION:

ENTRYCOUNT is a readonly attribute which reflects the value of the ENTRYCOUNT

attribute of the tntCwntuple HID. Readonly attributes cannot be changed
from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

HID - HBOOK ID for CWNtuple.
- denoting an object implementing the tntCwntuple interface.

RETURN:

The current value of ENTRYCOUNT is pushed onto the STAF_RESULT stack
(see SOC).

EXAMPLES:

EG1. Show the current value of the ENTRYCOUNT attribute of
tntCwntuple 100.

StAF TNT/CWNTUPLE/ENTRYCOUNT 100

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by HID can be found which
implements the tntCwntuple interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

8.5 /TNT/CWNTUPLE/HID HID

	name	description	type	default/range
required	HID	HBOOK ID for CWNtuple.	integer	no default

Get the HID attribute of the tntCwntuple HID. Useful?

DESCRIPTION:

HID is a readonly attribute which reflects the value of the HID
attribute of the tntCwntuple HID. Readonly attributes cannot be changed
from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

HID - HBOOK ID for CWNtuple.
- denoting an object implementing the tntCwntuple interface.

RETURN:

The current value of HID is pushed onto the STAF_RESULT stack
(see SOC).

EXAMPLES:

EG1. Show the current value of the HID attribute of
tntCwntuple 100.

StAF TNT/CWNTUPLE/HID 100

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by HID can be found which
implements the tntCwntuple interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

8.6 /TNT/CWNTUPLE/IMPORT HID TABLE

	name	description	type	default/range
required	HID	HBOOK ID for CWNtuple.	integer	no default
required	TABLE	tdmTable name	character string	no default

Load the contents of a table into an existing Ntuple.

DESCRIPTION:

IMPORT is a member function of objects which implement the tntCwntuple
interface. Though the name seems backwards, it loads a table into an
existing ntuple, not the reverse. Previous contents of the ntuple are
replaced. The ntuple definition and the table type must match.

ARGUMENTS:

HID - HBOOK ID for CWNtuple.
- denoting an ntuple object.

TABLE - tdmTable name.
- A StAF table of a type whose columns match those of the ntuple.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
tntCwntuple::IMPORT
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Replace contentes of ntuple 20 with table "bob"

StAF TNT/CWNTUPLE/IMPORT 20 bob

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by HID can be found which
implements the tntCwntuple interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

If the specified table does not exist, IMPORT is a quiet no-op.
Similarly, if the table type is wrong, no message, no action.
Despite the Exception advertised above, if the ntuple doesn't
exist, no message appears.

SEE ALSO:
TNT/NEWCWNTUPLE, TNT/CWNTUPLE/APPEND

8.7 /TNT/CWNTUPLE/TITLE HID

	name	description	type	default/range
required	HID	HBOOK ID for CWNtuple.	integer	no default

Get the TITLE attribute of the tntCwntuple HID.

DESCRIPTION:

TITLE is a readonly attribute which reflects the value of the TITLE attribute of the tntCwntuple HID. Readonly attributes cannot be changed from the user interface.

NB. Readonly attributes are not necessarily static attributes.

ARGUMENTS:

HID - HBOOK ID for CWNtuple.
- denoting an object implementing the tntCwntuple interface.

RETURN:

The current value of TITLE is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current value of the TITLE attribute of tntCwntuple 100.

StAF TNT/CWNTUPLE/TITLE 100

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by HID can be found which implements the tntCwntuple interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

8.8 /TNT/LIST

List all currently registered TNT worker objects.

DESCRIPTION:

Show in a table a one-line description for each TNT worker object currently registered with the TNT object factory.

The one-line description for each object is the result of an invokation of that object's listing method. The typical content of this listing is:
0 OID

The object's OID attribute (see SOC) presented as "%5d".
1 Lock State
The object's LOCK attribute (see SOC) presented as the divider character between the OID column and the NAME:OBJECT column. An object whose LOCK attribute is TRUE (cannot be deleted) uses the "-" character, whereas an object whose LOCK attribute is FALSE (can be deleted) uses "—" character.
2 NAME:OBJECT
The object's NAME attribute (see SOC) presented as "%-15s". Object names longer than 15 characters are abbreviated with a " " character at midpoint.
An object name is synonymous with an object instance.
3 TYPE:CLASS
The object's TYPE attribute (see SOC) presented as "%-15s". Object types longer than 15 characters are abbreviated with a " " character at midpoint.
An object type is synonymous with an object class.
4 DESCRIPTION
A class-specific description of the object.
for tntCWNtuples this is a blank.

TNT worker objects so far include only tntCWNtuple:

ARGUMENTS:

None.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the tntFactory::list() method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. List all currently registered TNT worker objects.

staf++ tnt/list
+——-+—————+—————+—————
— IDREF — NAME:OBJECT — TYPE:CLASS — DESCRIPTION
+——-+—————+—————+—————
— 105 — tntCWNtuple100 — tntCWNtuple —

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

Ntuple/list

8.9 /TNT/NEWCWNTUPLE HID TABLE

	name	description	type	default/range
required	HID	HBOOK ID for CWNtuple.	integer	no default
required	TABLE	tdmTable name	character string	no default

Create a new tntCwntuple object. Fill it with the current TABLE contents.

DESCRIPTION:

Each tntCwntuple created by the tntFactory shows up as an object managed by the tntFactory (see TNT/COUNT and TNT/LIST) and registered with the socCatalog (see SOC/COUNT and SOC/LIST).
The names of the HBOOK ntuple columns are those of the table columns. Each row of the table produces an entry in the Column Wise Ntuple.

ARGUMENTS:

HID - HBOOK ID of the Ntuple.
- Use this HID to specify this particular tntCwntuple object in subsequent commands.
- Currently HBOOK ntuple ID's are integers.

TABLE - tdmTable name.
- Name of the StAF table to be loaded into the HBOOK ntuple.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the tntFactory::newCwntuple
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Create a new tntCwntuple from table "bob"

StAF TNT/NEWCWNTUPLE 100 bob

EXCEPTIONS:

OBJECT_NOT_CREATED - The object creation failed. See error stack for detailed explanation of failure.

BUGS:

None known.

SEE ALSO:

TNT/CWNTUPLE

---

9. TOP, Table OPerators

9.1 /TOP/ARITHMETIC/OPERATE TABLE COLUMN OPERATION VALUE

	name	description	type	default/range
required	TABLE	Name of table	character string	no default
required	COLUMN	Name of column	character string	no default
required	OPERATION	Either add, multiply, divide, or subtract	character string	no default
required	VALUE	Value to apply	character string	no default

Simple arithmetic operations on tables.
Usage is almost self-explanatory.

Example:

TOP/ARITHMETIC/OPERATE /dui/tpc/tphit id subtract 1

This would subtract 1 from all the values in the id column.

9.2 /TOP/COUNT

Show the current count of TOP worker objects.

DESCRIPTION:

COUNT is a readonly long attribute which reflects the number of TOP worker objects currently registered with the TOP object factory. Constructing a new TOP worker object increments COUNT by 1, destroying an existing TOP worker object decrements COUNT by 1.

TOP worker objects include:
topCut - See TOP/CUT_AGENT
- An operator object for selecting rows from a table.
topJoin - See TOP/JOIN_AGENT
- An operator object for joining row-by-row data from two tables.
topProject - See TOP/PROJECT_AGENT
- An operator object for selecting columns from a table.
topSort - See TOP/SORT_AGENT
- An operator object for reordering rows in a table.

ARGUMENTS:

None.

RETURN:

The current value of COUNT is pushed onto the STAF_RESULT stack (see SOC).

EXAMPLES:

EG1. Show the current count of TOP worker objects.

StAF TOP/COUNT
TOP: Object count = 18

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

9.3 /TOP/CUT_AGENT/CUT SOREF TABLE [CUTFUNC]

	name	description	type	default/range
required	SOREF	topCut object SORef	character string	no default
required	TABLE	tdmTable name	character string	no default
optional	CUTFUNC	Cut function	character string	D='.'

Eliminate rows from a table which fail the cut function.

DESCRIPTION:

If FUNC is not specified:

Using the cut previously specified with the TOP/NEWCUT command,
remove all the rows from TABLE1 which do not pass the cut.

If FUNC is specified:

Remove all the rows from TABLE1 which do not
pass the cut, creating a new cut agent as a byproduct.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the topCut interface.

TABLE - tdmTable name.
- The table to be cut.

CUTFUNC - Cut function.
- A cut function with which to create a new topCut object.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
topCut::CUT
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Invoke the CUT method function of topCut "bob"
More guidance needed here.

StAF TOP/CUT_AGENT/CUT bob

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the topCut interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

9.4 /TOP/CUT_AGENT/FILTER SOREF TABLE1 TABLE2 [CUTFUNC]

	name	description	type	default/range
required	SOREF	topCut object SORef	character string	no default
required	TABLE1	tdmTable name of input table	character string	no default
required	TABLE2	tdmTable name of output table	character string	no default
optional	CUTFUNC	Cut function	character string	D='.'

Fill output table with rows from input table passing cut function.

DESCRIPTION:

If FUNC is not specified:

Using the cut previously specified with the TOP/NEWCUT command,
write a new table named TABLE2 using all the rows of TABLE1

that pass the cut.

If FUNC is specified:

Write a new table named TABLE2 using all the rows of TABLE1 that pass the cut function, creating a new cut agent as a byproduct.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the topCut interface.

TABLE1 - tdmTable name of input table.

TABLE2 - tdmTable name of output table.
- If TABLE2 does not exist, it will be created with the same columns TABLE1.

CUTFUNC - Cut function.
- A cut function with which to create a new topCut object.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the topCut::FILTER
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. FILTER rows from tb1 into tb2 where x 1000.

StAF TOP/CUT_AGENT/FILTER bigx tb1 tb2 x.gt.1000
More guidance needed here.

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the topCut interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

9.5 /TOP/CUT_AGENT/FUNCTION SOREF

	name	description	type	default/range
required	SOREF	topCut object SORef	character string	no default

Get the FUNCTION attribute of the topCut SOREF.

DESCRIPTION:

FUNCTION is a readonly attribute which defines the criterion upon which to select valid rows from a table. Readonly attributes cannot be changed from the user interface.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the topCut interface.

RETURN:

None.

EXAMPLES:

EG1. Show the current value of the FUNCTION attribute of
topCut "bob".

StAF TOP/CUT_AGENT/FUNCTION slowPions
pid.eq.5 .and. invpt.gt.1.12e3

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the topCut interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

9.6 /TOP/JOIN_AGENT/FASTJOIN SOREF TABLE1 TABLE2 TABLE3 [SELECT WHERE]

	name	description	type	default/range
required	SOREF	topJoin object SORef	character string	no default
required	TABLE1	tdmTable name of 1st input table	character string	no default
required	TABLE2	tdmTable name of 2nd input table	character string	no default
required	TABLE3	tdmTable name of output table	character string	no default
optional	SELECT	Selection specification	character string	D='-'
optional	WHERE	Where clause specification	character string	D='-'

Join two sorted tables to fill a third.

DESCRIPTION:

FASTJOIN is a member function of objects which implement the topJoin
interface.

You can read about joining in a book on relational databases
(eg, an SQL reference book).
You can use the same JOIN agent with either JOIN or FASTJOIN.
FASTJOIN works the same as JOIN, except as noted below.

It runs very fast (proportional to n instead of n squared).
Eg, two 10,000 row tables that previously took 20 minutes will
run in 0.12 seconds.

You must first sort each table on its corresponding
column in the WHERE clause of FASTJOIN.
You can sort the tables easily with TOP/SORT_AGENT/SORT.
This sort is based on a fancy sorting algorithm (quicksort),
and runs very fast.

The WHERE parameter must be simple, ie,
' ' column_name WHITE_SPACE column_name ' '.

No commas, no periods.
The first column_name is for the first table.
The second column_name is for the second table.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the topJoin interface.

TABLE1 - tdmTable name of 1st input table.

TABLE2 - tdmTable name of 2nd input table.

TABLE3 - tdmTable name of output table.
- Result of JOIN operation.
- If TABLE3 does not exist, it will be created with the
proper columns as defined by topJoin object SOREF.

SELECT - Selection specification.
- See TOP/PROJECT_AGENT for syntax of SELECT.
- If topJoin object SOREF does not exist, create it with
selection specification string SELECT.

WHERE - Where clause specification.
- See TOP/JOIN_AGENT for syntax of WHERE.
- If topJoin object SOREF does not exist, create it with
where clause string WHERE.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
topJoin::FASTJOIN
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Use an existing join object, j050 (defined by top/newjoin)
to join selected rows of the presorted ntphit and strack tables
into table newhit.
staf++ fastjoin j050 ntphit ProducedData/Tracks/strack newhit.
Real time 00:00:00, CP time 0.000

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the topJoin interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:
TOP/PROJECT_AGENT
TOP/JOIN_AGENT/JOIN

9.7 /TOP/JOIN_AGENT/JOIN SOREF TABLE1 TABLE2 TABLE3 [SELECT WHERE]

	name	description	type	default/range
required	SOREF	topJoin object SORef	character string	no default
required	TABLE1	tdmTable name of 1st input table	character string	no default
required	TABLE2	tdmTable name of 2nd input table	character string	no default
required	TABLE3	tdmTable name of output table	character string	no default
optional	SELECT	Selection specification	character string	D='-'
optional	WHERE	Where clause specification	character string	D='-'

Join two unsorted tables row-by-row to fill a third.

DESCRIPTION:

JOIN is a member function of objects which implement the topJoin interface.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the topJoin interface.

TABLE1 - tdmTable name of 1st input table.

TABLE2 - tdmTable name of 2nd input table.

TABLE3 - tdmTable name of output table.
- Result of JOIN operation.
- If TABLE3 does not exist, it will be created with the proper columns as defined by topJoin object SOREF.

SELECT - Selection specification.
- See TOP/PROJECT_AGENT for syntax of SELECT.
- If topJoin object SOREF does not exist, create it with selection specification string SELECT.

WHERE - Where clause specification.
- See TOP/JOIN_AGENT for syntax of WHERE.
- If topJoin object SOREF does not exist, create it with where clause specification string WHERE.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the topJoin::JOIN
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Use an existing join object, j050 (defined by top/newjoin) to join selected rows of the unsorted ntphit and strack tables into table newhit.

staf++ join j050 ntphit ProducedData/Tracks/strack newhit
Real time 00:00:00, CP time 0.010

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the topJoin interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

JOIN can be very slow for large tables. If your tables are sorted,
please use FASTJOIN.

SEE ALSO:
TOP/PROJECT
TOP/JOIN

9.8 /TOP/JOIN_AGENT/SELECTSPEC SOREF

	name	description	type	default/range
required	SOREF	topJoin object SORef	character string	no default

Get the SELECTSPEC attribute of the topJoin SOREF.

DESCRIPTION:

SELECTSPEC is a readonly attribute which defines the mapping of input table columns to output table columns for JOIN (or FASTJOIN) operations performed by the topJoin SOREF. Readonly attributes cannot be changed from the user interface.

SELECTSPEC is the SQL-like Selection Specification string for a topJoin object. See TOP/PROJECT_AGENT for syntax of Selection Specification strings.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the topJoin interface.

RETURN:

None.

EXAMPLES:

EG1. Show the current value of the SELECTSPEC attribute of topJoin "bob".

StAF TOP/JOIN_AGENT/SELECTSPEC bob
More guidance needed here.

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the topJoin interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:
TOP/PROJECT

9.9 /TOP/JOIN_AGENT/WHERECLAUSE SOREF

	name	description	type	default/range
required	SOREF	topJoin object SORef	character string	no default

Get the WHERECLAUSE attribute of the topJoin SOREF.

DESCRIPTION:

WHERECLAUSE is a readonly attribute which defines hows rows of two tables match for JOIN (or FASTJOIN) operations performed by topJoin SOREF. Readonly attributes cannot be changed from the user interface.

WHERECLAUSE is the SQL-like Where Clause string for a topJoin object. See TOP/JOIN_AGENT for syntax of Where Clause strings.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the topJoin interface.

RETURN:

None.

EXAMPLES:

EG1. Show the current value of the WHERECLAUSE attribute of topJoin "bob".

StAF TOP/JOIN_AGENT/WHERECLAUSE bob
More guidance needed here.

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the topJoin interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:
TOP/JOIN

9.10 /TOP/LIST

List all currently registered TOP worker objects.

DESCRIPTION:

Show a one-line description for each TOP worker object currently registered with the TOP object factory in a table for quick, simple perusal.

The one-line description for each object is the result of an invokation of that object's listing method. The typical content of this listing is:
0 OID
The object's OID attribute (see SOC) presented as "%5d".
1 Lock State
The object's LOCK attribute (see SOC) presented as the divider character between the OID column and the NAME:OBJECT column. An object whose LOCK attribute is TRUE (cannot be deleted) uses the "-" character, whereas an object whose LOCK attribute is FALSE (can be deleted) uses "—" character.

2 NAME:OBJECT
The object's NAME attribute (see SOC) presented as "%-15s".
Object names longer than 15 characters are abbreviated with a
" " character at midpoint.
An object name is synonymous with an object instance.
3 TYPE:CLASS
The object's TYPE attribute (see SOC) presented as "%-15s".
Object types longer than 15 characters are abbreviated with a
" " character at midpoint.
An object type is synonymous with an object class.
4 DESCRIPTION
A class-specific description of the object.
Sorts list nothing. Joins lists too much to fit the field.

TOP worker objects include:
topCut - See TOP/CUT_AGENT
- An operator object for selecting rows from a table.
topJoin - See TOP/JOIN_AGENT
- An operator object for joining row-by-row data from two tables.
topProject - See TOP/PROJECT_AGENT
- An operator object for selecting columns from a table.
topSort - See TOP/SORT_AGENT
- An operator object for reordering rows in a table.

ARGUMENTS:

None.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
topFactory::list()
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. List all currently registered TOP worker objects.

staf++ top/list
+—————————————————————
+——-+——————+——————+———————————-
— IDREF — NAME:OBJECT — TYPE:CLASS — DESCRIPTION
+——-+——————+——————+———————————-
— 58 — s025 — topSort —
— 59 — j025 — topJoin — tphit.id id, t# tphit.i
— 60 — s050 — topSort —
— 61 — s050a — topSort —
— 62 — j050 — topJoin — id, alpha, lam# ntphit.
+—————————————————————

EXCEPTIONS:

BUGS:

None known.

SEE ALSO:

9.11 /TOP/NEWCUT NAME CUTFUNC

	name	description	type	default/range
required	NAME	Name for new topCut object	character string	no default
required	CUTFUNC	Cut function	character string	no default

Create a new topCut object.

DESCRIPTION:

Each topCut created by the topFactory shows up as an object managed by the topFactory (see TOP/COUNT and TOP/LIST) and registered with the socCatalog (see SOC/COUNT and SOC/LIST).

This is the first step in making table cuts and/or filters.

ARGUMENTS:

NAME - Case-sensitive alphanumeric name for new topCut object.
- Use this name as part of SOREF (see SOC) to specify this particular topCut object in subsequent commands.

CUTFUNC - Cut function.
- A FORTRAN-like string defining a criterion on a table row.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
topFactory::newCut
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Create a new topCut.

StAF TOP/NEWCUT slowPions pid.eq.5.and.invpt.gt.1.12e3

In this case 'slowPions' is the name of the topCut object.
You will need this name in the second step.
The identifiers 'pid' and 'invpt' are column names.

For the second step, use either TOP/CUT_AGENT/FILTER or TOP/CUT_AGENT/CUT.

EXCEPTIONS:

OBJECT_NOT_CREATED - The object creation failed. See error stack for detailed explaination of failure.

BUGS:

None known.

SEE ALSO:

TOP/CUT_AGENT

9.12 /TOP/NEWJOIN NAME [SELECT WHERE]

	name	description	type	default/range
required	NAME	Name for new topJoin object	character string	no default
optional	SELECT	Selection specification	character string	D='-'

optional	WHERE	Where clause specification	character string	D='-'

Create a new topJoin object.

DESCRIPTION:

Each topJoin created by the topFactory shows up as an object managed by the topFactory (see TOP/COUNT and TOP/LIST) and registered with the socCatalog (see SOC/COUNT and SOC/LIST).

ARGUMENTS:

NAME - Case-sensitive alphanumeric name for new topJoin object.
- Use this name as part of SOREF (see SOC) to specify this particular topJoin object in subsequent commands.
- More guidance needed here.

SELECT - Selection specification.
- See TOP/PROJECT_AGENTfor syntax of SELECT.

WHERE - Where clause specification.
- See TOP/JOIN_AGENT for syntax of WHERE.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
topFactory::newJoin
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Create a new topJoin with NAME "j050" which will join 16 columns of the ntphit and strack tables where ntphit.track matches strack.trk

STAF[1] newjoin j050 ' id, alpha, lambda, row, x, y, z, track, cluster, _
STAF[1]_ q, xave, sigma, skew, kurto, npnt, chisqxy, chisqz ' _
STAF[1]_ ' ntphit.track strack.trk '

EXCEPTIONS:

OBJECT_NOT_CREATED - The object creation failed. See error stack for detailed explanation of failure.

BUGS:

If a join of the name given already exists newjoin fails w/o any message.

SEE ALSO:

TOP/JOIN_AGENT

9.13 /TOP/NEWPROJECT NAME [SELECT]

	name	description	type	default/range
required	NAME	Name for new topProject object	character string	no default
optional	SELECT	Selection specification	character string	D='-'

Create a new topProject object.

DESCRIPTION:

Each topProject created by the topFactory shows up as an object

managed by the topFactory (see TOP/COUNT and TOP/LIST) and
registered with the socCatalog (see SOC/COUNT and SOC/LIST).

ARGUMENTS:

NAME - Case-sensitive alphanumeric name for new topProject object.
- Use this name as part of SOREF (see SOC) to specify this particular
topProject object in subsequent commands.
- More guidance needed here.

SELECT - Selection specification.
- See TOP/PROJECT_AGENT for syntax of SELECT.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
topFactory::newProject
method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Create a new topProject with NAME "bob"

StAF TOP/NEWPROJECT bob

EXCEPTIONS:

OBJECT_NOT_CREATED - The object creation failed. See error stack for
detailed explaination of failure.

BUGS:

None known.

SEE ALSO:

TOP/PROJECT

9.14 /TOP/NEWSORT NAME COLUMN

	name	description	type	default/range
required	NAME	Name for new topSort object	character string	no default
required	COLUMN	Column name upon which to sort	character string	no default

Create a new topSort object to sort rows of a table.

DESCRIPTION:

Each topSort created by the topFactory shows up as an object
managed by the topFactory (see TOP/COUNT and TOP/LIST) and
registered with the socCatalog (see SOC/COUNT and SOC/LIST).

ARGUMENTS:

NAME - Case-sensitive alphanumeric name for new topSort object.
- Use this name as part of SOREF (see SOC) to specify this particular
topSort object in subsequent commands.

COLUMN - Column name upon which to sort.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
topFactory::newSort

method is pushed onto the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Create a new topSort which will sort tables on column "id".

StAF TOP/NEWSORT bob id

EXCEPTIONS:

OBJECT_NOT_CREATED - The object creation failed. See error stack for detailed explaination of failure.

BUGS:

None known.

SEE ALSO:

TOP/SORT_AGENT

9.15 /TOP/PROJECT_AGENT/PROJECT SOREF TABLE1 TABLE2 [SELECT]

	name	description	type	default/range
required	SOREF	topProject object SORef	character string	no default
required	TABLE1	tdmTable name of input table	character string	no default
required	TABLE2	tdmTable name of output table	character string	no default
optional	SELECT	Selection specification	character string	D='-'

Project columns of one table onto another.

DESCRIPTION:

PROJECT is a member function of objects which implement the topProject interface.

More guidance needed here.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the topProject interface.

TABLE1 - tdmTable name of input table.

TABLE2 - tdmTable name of output table.
Slashes are not allowed in this name, as a workaround
for a certain Staf bug.
- If TABLE2 does not exist, it will be created with the
proper columns as defined by topProject object SOREF.

SELECT - Selection specification.
- See TOP/PROJECT_AGENT for syntax of SELECT.
- If topProject object SOREF does not exist, create it with
selection specification string SELECT.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
topProject::PROJECT
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. More guidance needed here.

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the topProject interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

Staf mishandles the name of the output table if it contains
slashes. The logic is complex, and the safest fix I can think
of is to make slashes in such names illegal. I will do this
in July 98. Herb.

SEE ALSO:

9.16 /TOP/PROJECT_AGENT/SELECTSPEC SOREF

	name	description	type	default/range
required	SOREF	topProject object SORef	character string	no default

Get the SELECTSPEC attribute of the topProject SOREF.

DESCRIPTION:

SELECTSPEC is a readonly attribute which defines the mapping of input
table columns to output table columns for PROJECT operations performed
by the topProject SOREF. Readonly attributes cannot be changed from the
user interface.

SELECTSPEC is the SQL-like Selection Specification string for a
topProject object. See TOP/PROJECT_AGENT for syntax of Selection
Specification strings.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the topProject interface.

RETURN:

None.

EXAMPLES:

EG1. Show the current value of the SELECTSPEC attribute of
topProject "bob".

StAF TOP/PROJECT_AGENT/SELECTSPEC bob
More guidance needed here.

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the topProject interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:
TOP/PROJECT

9.17 /TOP/SORT_AGENT/COLUMN SOREF

	name	description	type	default/range
required	SOREF	topSort object SORef	character string	no default

Get the COLUMN attribute of the topSort SOREF.

DESCRIPTION:

COLUMN is a readonly attribute which defines the sorting order of SORT operations performed by the topSort object SOREF. Readonly attributes cannot be changed from the user interface.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the topSort interface.

RETURN:

None.

EXAMPLES:

EG1. Show the current value of the COLUMN attribute of topSort "bob".

StAF TOP/SORT_AGENT/COLUMN bob
More guidance needed here.

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which implements the topSort interface.
(See SOC/BIND to dynamically bind the proper resources, or rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

9.18 /TOP/SORT_AGENT/SORT SOREF TABLE

	name	description	type	default/range
required	SOREF	topSort object SORef	character string	no default
required	TABLE	tdmTable name	character string	no default

Sort table on column variable.

DESCRIPTION:

SORT reorders the rows in a table so that the column denoted by the COLUMN attribute of topSort object SOREF is sorted in ascending order.

ARGUMENTS:

SOREF - Stringified Object REFerence (see SOC).
- denoting an object implementing the topSort interface.

TABLE - tdmTable name.

RETURN:

Success (STAFCV_OK) or failure (STAFCV_BAD) of the
topSort::SORT
method is pushed on the STAF_STATUS stack (see SOC).

EXAMPLES:

EG1. Use the sort agent s025, defined by TOP/NEWSORT to sort the
table named.

staf++ TOP/SORT_AGENT/SORT s025 ProducedData/Hits/tphitau
number of rows is 71.

EXCEPTIONS:

OBJECT_NOT_FOUND - No object specified by SOREF can be found which
implements the topSort interface.
(See SOC/BIND to dynamically bind the proper resources, or
rebuild executable with the proper resources statically linked.)

BUGS:

None known.

SEE ALSO:

Index

Everything is converted to lower case before insertion into this index.