Quick Reference Guide

Adaptive Server Enterprise
12.5

Document ID: 70202-01-1250-04

Last revised: August 2001

Sybase, Inc., 6475 Christie Avenue, Emeryville, CA 94608.

Contents

Datatypes

Datatypes	Synonyms	Range
Exact numeric datatypes		
tinyint		0 to 255
smallint		-2^{15} (-32,768) to 215^{-1} (32,767)
int	integer	-2^{31} (-2,147,483,648) to 2^{31} -1 (2,147,483,647) bytes
numeric (*p*, *s*)		-10^{38} to 10^{38} -1
decimal (*p*, *s*)	dec	-10^{38} to 10^{38} -1
Approximate numeric datatypes		
float (*precision*)		Machine dependent
double precision		Machine dependent
real		Machine dependent
Money datatypes		
smallmoney		-214,748.3648 to 214,748.3647
money		-922,337,203,685,477.5808 to 922,337,203,685,477.5807
Date/time datatypes		
smalldatetime		January 1, 1900 to June 6, 2079
datetime		January 1, 1753 to December 31, 9999
Character datatypes		
char(*n*)	character	255 characters or fewer
varchar(*n*)	char[acter] varying	255 characters or fewer
nchar(*n*)	national char[acter]	255 characters or fewer
nvarchar(*n*)	nchar varying, national char[acter] varying	255 characters or fewer
Binary datatypes		
binary(*n*)		255 bytes or fewer
varbinary(*n*)		255 bytes or fewer
Bit datatype		
bit		0 or 1
Text and image datatype		
text		2^{31} -1 (2,147,483,647) bytes or fewer
image		2^{31} -1 (2,147,483,647) bytes or fewer

Reserved Words

Transact-SQL reserved words

A
add, all, alter, and, any, arith_overflow, as, asc, at, authorization, avg

B
begin, between, break, browse, bulk, by

C
cascade, case, char_convert, check, checkpoint, close, clustered, coalesce, commit, compute, confirm, connect, constraint, continue, controlrow, convert, count, create, current, cursor

D
database, dbcc, deallocate, declare, default, delete, desc, disk distinct, double, drop, dummy, dump

E
else, end, endtran, errlvl, errordata, errorexit, escape, except, exclusive, exec, execute, exists, exit, exp_row_size, external

F
fetch, fillfactor, for, foreign, from

G
goto, grant, group

H
having, holdlock

I
identity, identity_gap, identity_insert, identity_start, if, in, index, insert, install, intersect, into, is, isolation

J
jar, join

K
key, kill

L
level, like, lineno, load, lock

M
max, max_rows_per_page, min, mirror, mirrorexit, modify

N
national, noholdlock, nonclustered, not, null, nullif, numeric_truncation

O
of, off, offsets, on, once, online, only, open, option, or, order, over

P
partition, perm, permanent, plan, precision, prepare, primary, print, privileges, proc, procedure, processexit, proxy_table, public

Q
quiesce

R
raiserror, read, readpast, readtext, reconfigure, references remove, reorg, replace, replication, reservepagegap, return, revoke, role, rollback, rowcount, rows, rule

S
save, schema, select, set, setuser, shared, shutdown, some, statistics, stripe, sum, syb_identity, syb_restree

T
table, temp, temporary, textsize, to, tran, transaction, trigger, truncate, tsequal

U
union, unique, unpartition, update, use, user, user_option, using

V
values, varying, view

W
waitfor, when, where, while, with, work, writetext

SQL92 reserved words

A
absolute, action, allocate, are, assertion

B
bit, bit_length, both

C
cascaded, case, cast, catalog, char, char_length, character, character_length, coalesce, collate, collation, column, connection, constraints, corresponding, cross, current_date, current_time, current_timestamp, current_user

D
date, day, dec, decimal, deferrable, deferred, describe, descriptor, diagnostics, disconnect, domain

E
end-exec, exception, extract

F
false, first, float, found, full

G
get, global, go

H
hour

I
immediate, indicator, initially, inner, input, insensitive, int, integer, interval

J
join

L
language, last, leading, left, local, lower

M
match, minute, module, month

N
names, natural, nchar, next, no, nullif, numeric

O
octet_length, outer, output, overlaps

P
pad, partial, position, preserve, prior

R
real, relative, restrict, right

S
scroll, second, section, session_user, size, smallint, space, sql, sqlcode, sqlerror, sqlstate, substring, system_user

T
then, time, timestamp, timezone_hour, timezone_minute, trailing, translate, translation, trim, true

U
unknown, upper, usage
V
value, varchar
W
when, whenever, write, year
Z
zone

Potential SQL92 reserved words

A
after, alias, async
B
before, boolean, breadth
C
call, completion, cycle
D
data, depth, dictionary
E
each, elseif, equals
G
general
I
ignore
L
leave, less, limit, loop
M
modify
N
new, none
O
object, oid, old, operation, operators, others
P
parameters, pendant, preorder, private, protected
R
recursive, ref, referencing, resignal, return, returns, routine, row
S
savepoint, search, sensitive, sequence, signal, similar, sqlexception, structure
T
test, there, type
U
under
V
variable, virtual, visible
W
wait, without

Transact-SQL Functions

abs
abs(*numeric_expression*)

acos
acos(*cosine*)

ascii
ascii(*char_expr* | *uchar_expr*)

asin
asin(*sine*)

atan
atan(*tangent*)

atn2
atn2(*sine*, *cosine*)

avg
avg([all | distinct] *expression*)

ceiling
ceiling(*value*)

char
char(*integer_expr*)

charindex
charindex(*expression1*, *expression2*)

char_length
char_length(*char_expr* | uchar_expr)

col_length
col_length(*object_name*, *column_name*)

col_name
col_name(*object_id*, *column_id*[, *database_id*])

compare
compare(*char_expression1* |*uchar_expression1*),
(*char_expression2* | *uchar_expression2*)
[, { *collation_name* | *collation_ID* }]

convert
convert(*datatype* [(length) |
(*precision*[, *scale*])]
[null | not null], *expression* [, *style*])

cos
cos(*angle*)

cot
cot(*angle*)

count
count([all | distinct] *expression*)

curunreservedpgs
```
curunreservedpgs( dbid, lstart, unreservedpgs )
```

data_pgs
```
data_pgs(object_id,
  { data_oam_pg_id | index_oam_pg_id } )
```

datalength
```
datalength(expression)
```

dateadd
```
dateadd(date_part, integer, date)
```

datediff
```
datediff(datepart, date1, date2)
```

datename
```
datename(datepart, date)
```

datepart
```
datepart(date_part, date)
```

db_id
```
db_id(database_name)
```

db_name
```
db_name( [database_id] )
```

degrees
```
degrees(numeric)
```

difference
```
difference( char_expr1 | uchar_expr1 ),
  ( char_expr2 | char_expr2 )
```

exp
```
exp(approx_numeric)
```

floor
```
floor(numeric)
```

getdate
```
getdate( )
```

hextoint
```
hextoint(hexadecimal_string)
```

host_id
```
host_id( )
```

host_name
```
host_name( )
```

index_col
```
index_col(object_name, index_id, key_#
  [, user_id] )
```

index_colorder
```
index_colorder(object_name, index_id, key_#
  [, user_id] )
```

inttohex
```
inttohex(integer_expression)
```

isnull
```
isnull(expression1, expression2)
```

is_sec_service_on
```
is_sec_service_on(security_service_nm)
```

lct_admin
```
lct_admin( { { "lastchance" | "logfull" },
  database_id
  | "reserve", {log_pages | 0 }
  | "abort", process-id[, database-id ] } )
```

license_enabled
```
license_enabled( "ase_server" | "ase_ha" |
  "ase_dtm" | "ase_java" | "ase_asm" )
```

log
```
log(approx_numeric )
```

log10
```
log10(approx_numeric )
```

lower
```
lower( char_expr | uchar_expr )
```

ltrim
```
ltrim( char_expr | uchar_expr )
```

max
```
max(expression)
```

min
```
min(expression)
```

mut_excl_roles
```
mut_excl_roles(role1, role2
  [ membership | activation ] )
```

object_id
```
object_id(object_name)
```

object_name
```
object_name(object_id[, database_id] )
```

patindex
```
patindex("%pattern%", char_expr | uchar_expr
  [, using { bytes | characters | chars } ] )
```

pi
```
pi( )
```

power
```
power(value, power)
```

proc_role
```
proc_role("role_name")
```

ptn_data_pgs
```
ptn_data_pgs(object_id, partition_id)
```

radians
```
radians(numeric)
```

rand
```
rand( [integer] )
```

replicate
```
replicate(char_expr |uchar_expr, integer_expr )
```

reserved_pgs
```
reserved_pgs(object_id, { doampg | ioampg } )
```

reverse
```
reverse(expression | uchar_expr )
```

right
```
right(expression, integer_expr)
```

role_contain
```
role_contain("role1", "role2")
```

role_id
```
role_id("role_name")
```

role_name
```
role_name(role_id )
```

round
```
round(number, decimal_places)
```

rowcnt
```
rowcnt(sysindexes.doampg)
```

rtrim
```
rtrim( char_expr | uchar_expr )
```

show_role
```
show_role( )
```

show_sec_services
```
show_sec_services( )
```

sign
```
sign(numeric)
```

sin
```
sin(approx_numeric)
```

sortkey
```
sortkey( char_expression | uchar_expression
  [, { collation_name | collation_ID } ] )
```

soundex
```
soundex( char_expr | uchar_expr )
```

space
```
space(integer_expr)
```

sqrt
```
sqrt(approx_numeric)
```

str
```
str(approx_numeric[, length [, decimal] ] )
```

stuff
```
stuff(char_expr1 | uchar_expr1, start, length,
  char_expr2 | uchar_expr2 )
```

substring
```
substring(expression, start, length)
```

sum
```
sum( [ all | distinct ] expression)
```

suser_id
```
suser_id( [server_user_name] )
```

suser_name
```
suser_name( [server_user_id] )
```

syb_sendmsg
```
syb_sendmsg ip_address, port_number, message
```

tan
```
tan(angle)
```

textptr
```
textptr(column_name)
```

textvalid
```
textvalid("table_name.column_name",
  textpointer)
```

tsequal
```
tsequal(browsed_row_timestamp,
  stored_row_timestamp)
```

upper
```
upper(char_expr)
```

used_pgs
```
used_pgs(object_id, doampg, ioampg)
```

user
```
user
```

user_id
```
user_id( [user_name] )
```

user_name
```
user_name( [user_id] )
```

valid_name
```
valid_name(character_expression)
```

valid_user
```
valid_user(server_user_id)
```

Transact-SQL Commands

alter database

```
alter database database_name
  [on { default | database_device } [= size]
    [, database_device [ = size ] ] ... ]
  [ log on { default|database_device } [= size]
    [, database_device [ = size ] ] ... ]
  [with override]
  [for load]
  [for proxy_update]
```

alter role

```
alter role role1 { add | drop } exclusive
  { membership | activation } role2
alter role role_name [add passwd "password " |
  drop passwd] [lock | unlock]
alter role { role_name | "all overrides" }
  set { passwd expiration | min passwd length |
  max failed_logins } option_value
```

alter table

```
alter table [database.[owner].]table_name
  { add column_name datatype
    [ default {constant_expression|user|null } ]
    { identity | null | not null }
      [ off row | in row ]
    [ [constraint constraint_name]
      { { unique | primary key }
        [ clustered | nonclustered ] [ asc|desc ]
        [with { fillfactor = pct,
            max_rows_per_page = num_rows,
            reservepagegap = num_pages }]
        [on segment_name]
      | references [[database.]owner.]ref_table
        [ (ref_column) ]
      | check (search_condition) ] ...}
    [, next_column]...
  | add { [constraint constraint_name]
    { unique | primary key}
      [ clustered | nonclustered ]
      (column_name [ asc | desc ]
        [, column_name [ asc | desc ]...] )
      [with { fillfactor = pct,
          max_rows_per_page = num_rows,
          reservepagegap = num_pages}]
      [on segment_name]
    |foreign key (col_name[{,col_name}...])
      references [[database.]owner.]ref_table
        [ (ref_column [{, ref_column} ...])]
    | check (search_condition) }
  | drop {column_name [, column_name]...
          | constraint constraint_name }
  | modify col_name datatype [ null|not null ]
          [, next_column]...
  | replace column_name
     default { constant_expression|user|null }
  | partition number_of_partitions
  | unpartition
  | { enable | disable } trigger
  | lock { allpages | datarows | datapages } }
  | with exp_row_size=num_bytes
```

begin...end

```
begin
   statement block
  end
```

begin transaction

```
begin tran[saction] [transaction_name]
```

break

```
while logical_expression
   statement
  break
    statement
  continue
```

case

```
case
   when search_condition then expression
   [when search_condition then expression]...
   [else expression]
  end
```

case and values syntax:

```
case expression
   when expression then expression
   [when expression then expression]...
   [else expression]
  end
```

checkpoint

```
checkpoint
```

close

```
close cursor_name
```

coalesce

```
coalesce(expression, expression
  [, expression]...)
```

commit

```
commit [ tran | transaction ] | work ]
  [transaction_name]
```

compute Clause

```
start_of_select_statement
  compute row_aggregate ( column_name )
    [, row_aggregate( column_name )]...
  [by column_name [, column_name] ...]
```

connect to...disconnect

Component Integration Services only

```
connect to server_name
  disconnect
```

continue

```
while boolean_expression
   statement
  break
    statement
  continue
```

create database

```
create database database_name
  [on { default | database_device } [= size]
    [, database_device [ = size ] ]...]
  [log on database_device [ = size ]
    [, database_device [ = size ] ]...]
  [with {override|default_location =
    "pathname"}]
  [ for { load | proxy_update } ]
```

create default

```
create default [owner.]default_name
  as constant_expression
```

create existing table

Component Integration Services only

```
create existing table table_name (column_list)
  [ on segment_name ]
  [ [ external {table|procedure }] at pathname ]
```

create function

```
create function [owner.]sql_function_name
   ( [ sql_parameter_name sql_datatype
     [ (length) | (precision[, scale ] ) ]
   [ [, sql_parameter_name sql_datatype
     [ (length) | (precision[, scale ] ) ] ]
   ... ] ] )
  returns sql_datatype
    [ (length) | (precision[, scale ] ) ]
  [modifies sql data]
  [ returns null on null input|called on null
     input ]
  [ deterministic | not deterministic ]
  [exportable]
  language java
  parameter style java
  external name 'java_method_name
    [ ( [java_datatype[, java_datatype
    ...] ] ) ]'
```

create index

```
create [ unique ] [ clustered | nonclustered ]
     index index_name
  on [ [ database.]owner. ]table_name
     ( column_name [ asc | desc ]
     [, column_name [ asc | desc ] ]...)
  [ with { fillfactor = pct,
     max_rows_per_page = num_rows,
     reservepagegap = num_pages,
     consumers = x, ignore_dup_key, sorted_data,
     [ ignore_dup_row | allow_dup_row ],
     statistics using num_steps values } ]
  [ on segment_name ]
```

create plan

```
create plan query plan
    [ into group_name ]
    [ and set @new_id ]
```

create procedure

```
create procedure
  [owner.]procedure_name[;number]
  [ [ ( ] @parameter_name
    datatype [(length) | (precision [, scale])]
    [ = default ][ output ]
  [,@parameter_name
    datatype [ (length)|(precision [, scale])]
    [ = default ][ output ] ]...[ ) ] ]
  [ with recompile ]
  as {SQL_statements | external name dll_name}
```

create procedure (SQLJ)

```
create procedure [owner.]sql_procedure_name
  ( [ [in | out | inout] sql_parameter_name
    sql_datatype [ (length) |
    (precision[, scale]) ]
  [, [ in | out | inout ] sql_parameter_name
    sql_datatype [ (length) |
    (precision[, scale ]) ...])
  [ modifies sql data ]
  [ dynamic result sets integer ]
  [ deterministic | not deterministic ]
  language java
```

```
  parameter style java
  external name 'java_method_name
    [ ( [java_datatype[, java_datatype...]] ) ]'
```

create proxy_table

Component Integration Services only

```
create proxy_table table_name
  [ external table ] at pathname
```

create role

```
create role role_name [ with passwd "password"
  [, {"passwd expiration"|"min passwd length"|
    "max failed_logins"} option_value ] ]
```

create rule

```
create rule [[ and | or ] access]
  [owner.]rule_name
  as condition_expression
```

create schema

```
create schema authorization authorization_name
  create_oject_statement
    [ create_object_statement ... ]
  [ permission_statement ... ]
```

create table

```
create table [database.[owner].]table_name
  (column_name datatype
    [ default {constant_expression|user|null}]
    { [ { identity | null | not null } ]
      [off row | [in row [(size_in_bytes)] ]
    [ [constraint constraint_name]
        { { unique | primary key }
      [ clustered | nonclustered ] [ asc | desc ]
      [with { fillfactor = pct,
            max_rows_per_page = num_rows, }
            reservepagegap = num_pages } ]
      [on segment_name]
      | references [ [database.]owner.]ref_table
        [ (ref_column) ]
      | check (search_condition) } ] }...
  | [constraint constraint_name]
    { { unique | primary key }
      [ clustered | nonclustered ]
      (column_name [ asc | desc ]
        [ {, column_name [ asc | desc ] }... ] )
      [with { fillfactor = pct
         max_rows_per_page = num_rows ,
         reservepagegap = num_pages } ]
      [on segment_name]
    |foreign key (col_name [{, col_name}... ])
     references [[database.]owner.]ref_table
       [ (ref_column [ {, ref_column}... ])]
    | check (search_condition) ... }
  [ {, { next_column|next_constraint }}... ])
  [ lock { datarows | datapages | allpages } ]
  [ with { max_rows_per_page = num_rows,
      exp_row_size = num_bytes,
      reservepagegap = num_pages,
      identity_gap = num_values } ]
  [ on segment_name ]
  [ [ external table ] at pathname ]
```

create trigger

```
create trigger [ owner.]trigger_name
  on [ owner.]table_name
  for { insert , update, delete }
  as SQL_statements
```

Or, using the if update clause:

```
create trigger [owner.]trigger_name
    on [owner.]table_name
    for { insert , update }
    as
      [if update (column_name)
        [ { and|or } update (column_name)]...]
       SQL_statements
      [if update (column_name)
        [ { and|or } update (column_name)]...
       SQL_statements]...
```

create view

```
create view [owner.]view_name
  [ (column_name [, column_name]...) ]
  as select [distinct] select_statement
  [with check option]
```

dbcc

```
dbcc checkalloc [ (db_name [, fix | nofix ] ) ]
dbcc checkcatalog [ (db_name) ]
dbcc checkdb [ (db_name [, skip_ncindex] ) ]
dbcc checkstorage [ (db_name) ]
dbcc checktable({table_name|table_id}
  [, skip_ncindex])
dbcc checkverify [ (db_name) ]
dbcc complete_xact (xid, { "commit" |
  "rollback" } )
dbcc forget_xact (xid)
dbcc dbrepair (db_name, dropdb)
dbcc engine({offline, [enginenum]| "online"})
dbcc fix_text ( { table_name | table_id } )
dbcc indexalloc ( { table_name | table_id },
  index_id [, { full | optimized | fast | null }
  [, fix | nofix] ] )
dbcc rebuild_text (table [, column
  [, text_page_number] ] )
dbcc reindex ( { table_name | table_id } )
dbcc tablealloc ( { table_name | table_id }
    [, { full | optimized | fast | null }
    [, fix | nofix] ] ) |
dbcc { traceon|traceoff } (flag [, flag ... ] )
dbcc tune ( { ascinserts, {0|1} , tablename |
     cleanup, { 0 | 1 } |
     cpuaffinity, start_cpu {, on| off } |
     des_greedyalloc, dbid, object_name,
       " { on | off }" |
     deviochar vdevno, "batch_size" |
     doneinproc { 0 | 1 } |
     maxwritedes, writes_per_batch } )
```

deallocate cursor

```
deallocate cursor cursor_name
```

declare

Variable declaration:

```
declare @variable_name datatype
  [, @variable_name datatype]...
```

Variable assignment:

```
select @variable = { expression |
  select_statement} [, @variable = {expression |
  select_statement } ...] [from table_list]
```

```
  [where search_conditions]
  [group by group_by_list]
  [having search_conditions]
  [order by order_by_list]
  [compute function_list [by by_list] ]
```

declare cursor

```
declare cursor_name cursor
  for select_statement
  [for {read only|update [of col_name_list]}]
```

delete

```
delete [from]
  [[database.]owner.]{ view_name | table_name }
  [where search_conditions]
  [plan "abstract plan"]
delete [ [db.]owner.]{table_name | view_name }
  [from [ [db.]owner.]{view_name [ readpast ] |
    table_name [readpast]
      [ (index {index_name | table_name }
      [ prefetch size ] [lru|mru] ) ] }
  [, [ [database.]owner.]{view_name [readpast] |
    table_name [readpast]
      [ (index {index_name | table_name }
      [ prefetch size ] [ lru | mru ])]} ...]
  [where search_conditions] ]
  [plan "abstract plan"]
delete [from]
  [ [database.]owner.]{ table_name | view_name }
  where current of cursor_name
```

delete statistics

```
delete [shared] statistics table_name
  [(col_name
  [, col_name]...) ]
```

disk init

```
disk init
  name = "device_name" ,
  physname = "physicalname" ,
  [vdevno = virtual_device_number] ,
  size = number_of_device
  [, vstart = virtual_address]
    cntrltype = controller_number ]
  [, dsync = { true | false } ]
```

disk mirror

```
disk mirror
  name = "device_name" ,
  mirror = "physicalname"
  [, writes = { serial | noserial } ]
  [, contiguous ] (OpenVMS only)
```

disk refit

```
disk refit
```

disk reinit

```
disk reinit
  name = "device_name",
  physname = "physicalname" ,
  [vdevno = virtual_device_number] ,
  size = size_of_device
  [, vstart = virtual_address]
  [, dsync = { true | false }
```

disk remirror

```
disk remirror name = "device_name"
```

disk unmirror

```
disk unmirror
  name = "device_name"
  [, side = { "primary" | secondary } ]
  [, mode = { retain | remove } ]
```

drop database

```
drop database database_name
  [, database_name]...
```

drop default

```
drop default [owner.]default_name
  [, [owner.]default_name]...
```

drop function

```
drop func[tion] [owner.]function_name
  [, [owner.]function_name] ...
```

drop index

```
drop index table_name.index_name
  [, table_name.index_name]...
```

drop procedure

```
drop proc[edure] [owner.]procedure_name
  [, [owner.]procedure_name]...
```

drop role

```
drop role role_name [with override]
```

drop rule

```
drop rule [owner.]rule_name
  [, [owner.]rule_name]...
```

drop table

```
drop table [ [database.]owner.]table_name
  [, [ [database.]owner.]table_name ]...
```

drop trigger

```
drop trigger [owner.]trigger_name
  [, [owner.]trigger_name]...
```

drop view

```
drop view [owner.]view_name
  [, [owner.]view_name]...
```

dump database

```
dump database database_name
  to
  [compress::[compress_level::]]stripe_device
    [at backup_server_name]
    [density = density_value,
    blocksize = number_bytes,
    capacity = number_kilobytes,
    dumpvolume = volume_name,
    file = file_name]
  [stripe on
    [compress::[compress_level::]]stripe_device
    [at backup_server_name]
    [density = density_value,
    blocksize = number_bytes,
    capacity = number_kilobytes,
    dumpvolume = volume_name,
    file = file_name] ]
  [[stripe on
    [compress::[compress_level::]stripe_device
    [at backup_server_name]
    [density = density_value,
    blocksize = number_bytes,
```

```
    capacity = number_kilobytes,
    dumpvolume = volume_name,
    file = file_name] ]... ]
  [with {
    density = density_value,
    blocksize = number_bytes,
    capacity = number_kilobytes,
    dumpvolume = volume_name,
    file = file_name,
    [ dismount | nodismount ],
    [ nounload | unload ],
    retaindays = number_days,
    [ noinit | init ],
    notify = {client|operator_console}}]
```

dump transaction

Making a routine log dump:

```
dump tran[saction] dbname
  to
  [compress::[compress_level::]]stripe_device
    [at backup_server_name]
    [density = density_value,
    blocksize = number_bytes,
    capacity = number_kilobytes,
    dumpvolume = volume_name,
    file = file_name]
  [stripe on
    [compress::[compressilevel::]]stripe_device
    [at backup_server_name]
    [density = density_value,
    blocksize = number_bytes,
    capacity = number_kilobytes,
    dumpvolume = volume_name,
    file = file_name] ]
  [ [stripe on
    [compress::[compress_level::]]stripe_device
    [at backup_server_name]
    [density = density_value,
    blocksize = number_bytes,
    capacity = number_kilobytes,
    dumpvolume = volume_name,
    file = file_name] ]...]
  [with {
    density = density_value,
    blocksize = number_bytes,
    capacity = number_kilobytes,
    dumpvolume = volume_name,
    file = file_name,
    [ dismount | nodismount ],
    [ nounload | unload ],
    retaindays = number_days,
    [ noinit | init ],
    notify = { client | operator_console },
    standby_access } ]
```

Trucating log without making backup copy:

```
dump tran[saction] dbname with truncate_only
```

Backing up log after database device fails:

```
dump tran[saction] dbname
  to
  [compress::[compress_level::]]stripe_device
    [density = density_value,
    blocksize = number_bytes,
    capacity = number_kilobytes,
    dumpvolume = volume_name,
    file = file_name]
  [stripe on
    [compress::[compress_level::]]stripe_device
    [at backup_server_name]
    [density = density_value,
    blocksize = number_bytes,
    capacity = number_kilobytes,
    dumpvolume = volume_name,
    file = file_name] ]
```

```
  [[stripe on
    [compress::[compress_level::]]stripe_device
    [at backup_server_name]
    [density = density_value,
    blocksize = number_bytes,
    capacity = number_kilobytes,
    dumpvolume = volume_name,
    file = file_name] ]...]
  [with {
    density = density_value,
    blocksize = number_bytes,
    capacity = number_kilobytes,
    dumpvolume = volume_name,
    file = file_name,
    [ dismount | nodismount ],
    [ nounload | unload ],
    retaindays = number_days,
    [ noinit | init ],
    no_truncate,
    notify = { client | operator_console } } ]
```

execute

```
[exec[ute] ] [@return_status = ]
  [[[server.]db.]owner.]procedure_name[;number]
     [ [@parameter_name =] value |
       [@parameter_name =] @variable [output]
     [, [@parameter_name =] value |
       [@parameter_name =] @variable
  [output]...] ]
  [with recompile]
  or
    exec[ute] ("string " | char_variable
  [+ "string" | char_variable ]...)
```

fetch

```
fetch cursor_name [ into fetch_target_list ]
```

goto label

```
label:
  goto label
```

grant

Granting permission to access database objects:

```
grant { all [privileges] | permission_list }
  on { table_name [ (column_list) ]
     | view_name[ (column_list) ]
     | stored_procedure_name}
  to { public | name_list | role_name }
  [with grant option]
```

Granting permission to execute certain commands:

```
grant { all [privileges] | command_list }
  to { public | name_list | role_name }
```

Granting a role to a user or a role:

```
grant {role role_granted [, role_granted ...]}
  to grantee [, grantee...]
```

group by and having Clauses

```
Start of select statement
  [group by [all] aggregate_free_expression
  [, aggregate_free_expression]...]
  [having search_conditions]
End of select statement
```

if...else

```
if logical_expression [plan "abstract plan"]
   statements
  [else
```

```
    [if logical_expression]
    [plan "abstract plan"]
  statement]
```

insert

```
insert [into]
  [database.[owner.]] {table_name|view_name}
  [ (column_list) ]
  { values (expression [, expression]...) |
    select_statement [plan "abstract plan"] }
```

kill

```
kill spid
```

load database

```
load database database_name
  from [compress::]stripe_device
    [at backup_server_name]
    [density = density_value,
    blocksize = number_bytes,
    dumpvolume = volume_name,
    file = file_name]
  [stripe on [compress::]stripe_device
    [at backup_server_name ]
    [density = density_value,
    blocksize = number_bytes,
    dumpvolume = volume_name,
    file = file_name]
  [ [stripe on [compress::]stripe_device
    [at backup_server_name ]
    [density = density_value,
    blocksize = number_bytes,
    dumpvolume = volume_name,
    file = file_name] ]...]
  [with {
    density = density_value,
    blocksize = number_bytes,
    dumpvolume = volume_name,
    file = file_name,
    [ dismount | nodismount ],
    [ nounload | unload ],
    listonly [= full],
    headeronly,
    notify = { client | operator_console }
    } ] ]
```

load transaction

```
load tran[saction] database_name
  from [compress::]stripe_device
    [at backup_server_name]
    [density = density_value,
    blocksize = number_bytes,
    dumpvolume = volume_name,
    file = file_name]
  [stripe on [compress::]stripe_device
    [at backup_server_name ]
    [density = density_value,
    blocksize = number_bytes,
    dumpvolume = volume_name,
    file = file_name]
  [ [stripe on [compress::]stripe_device
    [at backup_server_name ]
    [density = density_value,
    blocksize = number_bytes,
    dumpvolume = volume_name,
    file = file_name]]...]
  [with {
    density = density_value,
    blocksize = number_bytes,
    dumpvolume = volume_name,
    file = file_name,
    [dismount | nodismount],
```

```
    [nounload | unload],
    listonly [= full],
    headeronly,
    notify = { client | operator_console }
    until_time = datetime} ] ]
```

lock table

```
lock table table_name in {share|exclusive} mode
  [ wait [ numsecs ] | nowait ]
```

nullif

```
nullif(expression, expression)
```

online database

```
online database database_name [for
  standby_access]
```

open

```
open cursor_name
```

order by Clause

```
[Start of select statement]
  [order by {[table_name.| view_name.]col_name
    | select_list_number | expression} [ asc |
  desc ] [, { [table_name.| view_name.]col_name
      select_list_number|expression}
        [asc|desc]]...
  [End of select statement]
```

prepare transaction

```
prepare tran[saction]
```

print

```
print
  {format_string | @local_variable |
  @@global_variable}
     [, arg_list]
```

quiesce database

```
quiesce database tag_name hold dbname
  [, dbname]
  [for external dump]
quiesce database tag_name release
```

raiserror

```
raiserror error_number
  [{format_string | @local_variable}]
  [, arg_list]
  [with errordata restricted_select_list]
```

readtext

```
readtext [
  [database.]owner.]table_name.column_name
  text_pointer offset size
  [ holdlock | noholdlock ] [readpast]
  [ using {bytes | chars | characters} ]
  [at isolation {
    [ read uncommitted | 0 ] |
    [ read committed | 1 ] |
    [ repeatable read | 2 ] |
    [ serializable | 3 ] } ]
```

reconfigure

```
reconfigure
```

remove java

```
remove java
  class class_name [, class_name]...
  | package package_name [, package_name]...
  | jar jar_name [, jar_name]...[retain classes]
```

reorg

```
reorg reclaim_space tablename [indexname]
  [with { resume, time = no_of_minutes } ]
reorg forwarded_rows tablename
  [with { resume, time = no_of_minutes } ]
reorg compact tablename
  [with { resume, time = no_of_minutes } ]
reorg rebuild [ tablename | indexname ]
```

return

```
return [integer_expression] [plan "abstract
  plan"]
```

revoke

Revoking permissions to access database objects:

```
revoke [grant option for]
  { all [privileges] | permission_list }
  on { table_name [ (column_list) ]
     | view_name [ (column_list) ]
     | stored_procedure_name}
  from { public|name_list|role_name }
  [cascade]
```

Revoking permissions to create database objects, execute set proxy, or execute set session authorization:

```
revoke { all [privileges] | command_list }
  from { public | name_list | role_name }
```

Revoking a role from a user or another role:

```
revoke role {role_name [, role_name ...] }
  from {grantee [, grantee ...] }
```

rollback

```
rollback [ tran[saction] | work ]
  [ transaction_name | savepoint_name ]
```

rollback trigger

```
rollback trigger
  [with raiserror_statement]
```

save transaction

```
save transaction savepoint_name
```

select

```
select ::=
   select [ all | distinct ] select_list
   [into_clause]
   [from_clause]
   [where_clause]
   [group_by_clause]
   [having_clause]
   [order_by_clause]
   [compute_clause]
   [read_only_clause]
   [isolation_clause]
   [browse_clause]
   [plan_clause]
  select_list ::=
```

For details on *select_list*, see "Parameters" in the *Reference Manual*.

```
  into_clause ::=
    into [[database.]owner.]table_name
    [ lock { datarows | datapages | allpages } ]
    [ with into_option [, into_option] ...]
```

```
  into_option ::=
    | max_rows_per_page = num_rows
    | exp_row_size = num_bytes
    | reservepagegap = num_pages
    | identity_gap = gap
    [ existing table table_name ]
    [ [ external type ] at "path_name"
    [ column delimiter delimiter ] ]

from_clause ::=
  from table_reference [, table_reference]...

  table_reference ::=
    table_view_name | ANSI_join

    table_view_name ::=
      [[database.]owner.]
        { { table_name |view_name }
      [as] [correlation_name]
      [index { index_name | table_name } ]
      [parallel [degree_of_parallelism] ]
      [prefetch size] [ lru | mru ] }
    [holdlock | noholdlock ]
    [readpast]
    [shared]

    ANSI_join ::=
      table_reference join_type join
           table_reference join_conditions
        join_type ::= inner|left [outer]|right
           [outer]
        join_conditions ::= on search_conditions

where_clause ::=
  where search_conditions

group_by_clause ::=
  group by [all] aggregate_free_expression
    [, aggregate_free_expression]...

having_clause ::=
  having search_conditions

order_by_clause ::=
  order by sort_clause [, sort_clause]...

  sort_clause ::=
    { [ [ [database.]owner.]{ table_name.
      | view_name. } ]column_name
      | select_list_number | expression }
      [ asc | desc ]

compute_clause ::=
  compute row_aggregate(column_name)
    [, row_aggregate(column_name) ]...
  [by column_name [, column_name]...]

read_only_clause ::=
  for {read only | update [of col_name_list]}

isolation_clause ::=
  at isolation
      { read uncommitted | 0 }
      | { read committed | 1 }
      | { repeatable read | 2 }
      | { serializable | 3 }

browse_clause ::=
  for browse

plan_clause ::=
  plan "abstract plan"
```

set

```
set ansinull { on | off }
set ansi_permissions { on | off }
set arithabort [ arith_overflow |
  numeric_truncation ] { on | off }
set arithignore [arith_overflow] { on | off }
set {chained, close on endtran, nocount,
  noexec, parseonly, procid, self_recursion,
  showplan, sort_resources} {on | off}
set char_convert {off | on [with {error |
  no_error}] |
  charset [with { error | no_error } ] }
set cis_rpc_handling { on | off }
set [clientname client_name | clienthostname
  host_name | clientapplname application_name]
set cursor rows number for cursor_name
set {datefirst number, dateformat format,
  language language}
set fipsflagger { on | off }
set flushmessage { on | off }
set forceplan { on | off }
set identity_insert
  [database.[owner.]]table_name
  { on | off }
set jtc { on | off }
set lock { wait [ numsecs ] | nowait }
set offsets {select, from, order, compute,
  table, procedure, statement, param, execute}
  { on | off }
set parallel_degree number
set plan { dump|load } [group_name] { on|off }
set plan exists check { on | off }
set plan replace { on | off }
set prefetch [ on | off ]
set process_limit_action {abort|quiet|warning}
set proxy login_name
set quoted_identifier { on | off }
set role {"sa_role" | "sso_role" | "oper_role"
  | role_name [with passwd "password"]} {on|off}
set {rowcount number, textsize number}
set scan_parallel_degree number
set session authorization login_name
set sort_merge { on | off }
set statistics {io, subquerycache, time}
  {on|off}
set statistics simulate { on | off }
set strict_dtm_enforcement { on | off }
set string_rtruncation { on | off }
set stringsize number
set table count number
set textsize {number}
```

```
set transaction isolation level {
  [ read uncommitted | 0 ] |
  [ read committed | 1 ] |
  [ repeatable read | 2 ]|
  [ serializable | 3 ] }
set transactional_rpc { on | off }
```

setuser

```
setuser ["user_name"]
```

shutdown

```
shutdown [srvname] [with { wait | nowait } ]
```

truncate table

```
truncate table [ [database.]owner.]table_name
```

union

```
select select_list [into clause]
       [from clause] [where clause]
       [group by clause] [having clause]
  [union [all]
     select select_list
       [from clause] [where clause]
       [group by clause] [having clause] ]...
  [order by clause]
  [compute clause]
```

update

```
update [[db.]owner.]{table_name|view_name}
  set [[[db.]owner.]{table_name.|view_name.}]
     column_name1 =
       {expression1|NULL|(select_statement)} |
     variable_name1 =
       {expression1|NULL|(select_statement)}
     [, column_name2 =
       {expression2|NULL|
         (select_statement)}]...|
     [, variable_name2 =
       {expression2 | NULL |
         (select_statement) } ]...

  [from [ [db.]owner.]{view_name [readpast] |
     table_name [readpast]
       [ (index { index_name | table_name }
       [prefetch size][ lru | mru ] ) ] }
      [,[ [db.]owner.]{view_name [readpast] |
     table_name [readpast]
     [ (index { index_name | table_name }
     [prefetch size ][lru|mru] ) ] } ]
  ...]
  [where search_conditions]
  [plan "abstract plan"]
update [ [db.]owner.]{table_name|view_name}
  set [[[db.]owner.]{table_name.|view_name.}]
    column_name1 =
      {expression1 |NULL|(select_statement)}|
    variable_name1 =
      {expression1|NULL|(select_statement)}
    [, column_name2 =
      {expression2 |NULL|
        (select_statement)}]... |
    [, variable_name2 =
      {expression2 | NULL |
        (select_statement) } ]...
  where current of cursor_name
```

update all statistics

```
update all statistics table_name
```

update partition statistics

```
update partition statistics table_name
  [partition_number]
```

update statistics

```
update statistics table_name
  [ [ index_name] | [ (column_list) ] ]
  [using step values]
  [With consumers = consumers ]
update index statistics table_name [index_name]
  [using step values]
  [With consumers = consumers ]
```

use

```
use database_name
```

waitfor

```
waitfor { delay time | time time | errorexit |
  | processexit | mirrorexit }
```

where Clause

```
where [not] expression comparison_operator
  expression

where [not] expression [not] like
    "match_string"
  [escape "escape_character"]

where [not] expression is [not] null

where [not]
  expression [not] between expression and
  expression

where [not]
  expression [not] in ({value_list|subquery})

where [not] exists (subquery)

where [not]
  expression comparison_operator
    {any|all} (subquery)

where [not] col_name join_operator col_name

where [not] logical_expression

where [not] expression {and|or} [not]
  expression
```

while

```
while logical_expression
  [plan "abstract plan"] statement
```

writetext

```
writetext [[db.]owner.]table_name.col_name
  text_pointer [readpast] [with log] data
```

System Procedures

sp_activeroles
```
sp_activeroles [expand_down]
```

sp_addalias
```
sp_addalias loginame, name_in_db
```

sp_addauditrecord
```
sp_addauditrecord [text [, db_name [, obj_name
  [, owner_name [, dbid [, objid] ] ] ] ] ]
```

sp_addaudittable
```
sp_addaudittable devname
```

sp_addengine
```
sp_addengine engine_number, engine_group
```

sp_addexeclass
```
sp_addexeclass classname, priority, timeslice,
  engine_group
```

sp_addextendedproc
```
sp_addextendedproc esp_name, dll_name
```

sp_addexternlogin
Component Integration Services only
```
sp_addexternlogin server, loginname, externname
  [, externpassword]
```

sp_addgroup
```
sp_addgroup grpname
```

sp_addlanguage
```
sp_addlanguage language, alias, months,
  shortmons, days, datefmt, datefirst
```

sp_addlogin
```
sp_addlogin loginname, passwd [, defdb]
  [, deflanguage ] [, fullname ] [, passwdexp ]
  [, minpwdlen ] [, maxfailedlogins ]
```

sp_addmessage
```
sp_addmessage message_num, message_text [,
  language [, with_log [, replace] ] ]
```

sp_addobjectdef
Component Integration Services only
```
sp_addobjectdef tablename, "objectdef"
  [,"objecttype"]
```

sp_add_qpgroup
```
sp_add_qpgroup new_name
```

sp_addremotelogin
```
sp_addremotelogin remoteserver [, loginame
  [, remotename] ]
```

sp_add_resource_limit
```
sp_add_resource_limit name, appname, rangename,
  limittype, limitvalue [, enforced [, action
  [, scope ] ] ]
```

sp_addsegment
```
sp_addsegment segname, dbname, devname
```

sp_addserver
```
sp_addserver lname [, class [, pname] ]
```

sp_addthreshold
```
sp_addthreshold dbname, segname, free_space,
  proc_name
```

sp_add_time_range
```
sp_add_time_range name, startday, endday,
  starttime, endtime
```

sp_addtype
```
sp_addtype typename,
  phystype [ (length) | (precision [, scale] ) ]
  [, "identity" | nulltype ]
```

sp_addumpdevice
```
sp_addumpdevice { "tape" | "disk" },
  logicalname, physicalname [, tapesize ]
```

sp_adduser
```
sp_adduser loginame
  [, name_in_db [, grpname ] ]
```

sp_altermessage
```
sp_altermessage message_id, parameter,
  parameter_value
```

sp_audit
```
sp_audit option, login_name, object_name
  [, setting ]
```

sp_autoconnect
Component Integration Services only
```
sp_autoconnect server, { true | false }
  [, loginame]
```

sp_bindcache
```
sp_bindcache cachename, dbname
  [, [ownername.]tablename
  [, indexname | "text only"] ]
```

sp_bindefault
```
sp_bindefault defname, objname [, futureonly]
```

sp_bindexeclass
```
sp_bindexeclass "object_name", "object_type",
  "scope", "classname"
```

sp_bindmsg
```
sp_bindmsg constrname, msgid
```

sp_bindrule
```
sp_bindrule rulename, objname [, futureonly]
```

sp_cacheconfig
```
sp_cacheconfig [cachename
  [, "cache_size[P|K|M|G]"] [, logonly|mixed]
  [,strict | relaxed ] ]
  [, "cache_partition=[1|2|4|8|16|32|64]"]
```

sp_cachestrategy

```
sp_cachestrategy dbname, [ownername.]tablename
  [, indexname | "text only" | "table only"
  [, { prefetch | mru }, { "on" | "off" } ] ]
```

sp_changedbowner

```
sp_changedbowner loginame [, true ]
```

sp_changegroup

```
sp_changegroup grpname, username
```

sp_checknames

```
sp_checknames
```

sp_checkreswords

```
sp_checkreswords [user_name_param]
```

sp_checksource

```
sp_checksource [objname
  [, tabname [, username] ] ]
```

sp_chgattribute

```
sp_chgattribute objname, {"max_rows_per_page" |
  "fillfactor" | "reservepagegap" |
  "exp_row_size" concurrency_opt_threshold },
  optvalue
```

sp_clearpsexe

```
sp_clearpsexe spid, exeattr
```

sp_clearstats

```
sp_clearstats [loginame]
```

sp_cmp_all_qplans

```
sp_cmp_all_qplans group1, group2 [, mode]
```

sp_cmp_qplans

```
sp_cmp_qplans id1, id2
```

sp_commonkey

```
sp_commonkey tabaname, tabbname, col1a, col1b
  [, col2a, col2b, ..., col8a, col8b]
```

sp_companion

```
sp_companion
  [ server_name
  { ,configure
      [, { with_proxydb | NULL } ]
      [, srvlogin ]
      [, server_password ]
      [, cluster_login ]
      [, cluspassword ] ]
  | drop
  | suspend
  | resume
  | prepare_failback
  | do_advisory }
      {, all
      | help
      | group attribute_name
      | base attribute_name}
```

sp_configure

```
sp_configure [configname [, configvalue] |
  group_name | non_unique_parameter_fragment]
  ["p|P|k|K|m|M|g|G"]]
```

or

```
sp_configure "configuration file", 0, {"write"
    |"read"|"verify"|"restore"} "file_name"
```

sp_copy_all_qplans

```
sp_copy_all_qplans src_group, dest_group
```

sp_copy_qplan

```
sp_copy_qplan src_id, dest_group
```

sp_countmetadata

```
sp_countmetadata "configname" [, dbname]
```

sp_cursorinfo

```
sp_cursorinfo [{cursor_level|NULL}]
  [, cursor_name]
```

sp_dboption

```
sp_dboption [dbname, optname, { true|false } ]
```

sp_dbrecovery_order

```
sp_dbrecovery_order
  [database_name [, rec_order [, force] ] ]
```

sp_dbremap

```
sp_dbremap dbname
```

sp_defaultloc

Component Integration Services only

```
sp_defaultloc dbname, { "defaultloc" | NULL }
  [, "defaulttype"]
```

sp_depends

```
sp_depends objname
```

sp_deviceattr

```
sp_deviceattr logicalname, optname, optvalue
```

sp_diskdefault

```
sp_diskdefault logicalname,
  {defaulton|defaultoff}
```

sp_displayaudit

```
sp_displayaudit ["procedure"|"object"|"login"|
  "database" | "global" | "default_object" |
  "default_procedure" [, name] ]
```

sp_displaylevel

```
sp_displaylevel [loginame [, level] ]
```

sp_displaylogin

```
sp_displaylogin [loginame [, expand_up |
  expand_down ] ]
```

sp_displayroles

```
sp_displayroles [ grantee_name [, mode ] ]
```

sp_dropalias

```
sp_dropalias loginame
```

sp_drop_all_qplans

```
sp_drop_all_qplans name
```

sp_dropdevice
```
sp_dropdevice logicalname
```

sp_dropengine
```
sp_dropengine engine_number, engine_group
```

sp_dropexeclass
```
sp_dropexeclass classname
```

sp_dropextendedproc
```
sp_dropextendedproc esp_name
```

sp_dropexternlogin
Component Integration Services only
```
sp_dropexternlogin remote_server [, login_name]
```

sp_dropglockpromote
```
sp_dropglockpromote { "database" | "table" },
  objname
```

sp_dropgroup
```
sp_dropgroup grpname
```

sp_dropkey
```
sp_dropkey keytype, tabname [, deptabname ]
```

sp_droplanguage
```
sp_droplanguage language [, dropmessages ]
```

sp_droplogin
```
sp_droplogin loginame
```

sp_dropmessage
```
sp_dropmessage message_num [, language ]
```

sp_drop_qpgroup
```
sp_drop_qpgroup group
```

sp_drop_qplan
```
sp_drop_qplan id
```

sp_dropobjectdef
Component Integration Services only
```
sp_dropobjectdef "object_name"
```

sp_dropremotelogin
```
sp_dropremotelogin remoteserver [, loginame
  [, remotename] ]
```

sp_drop_resource_limit
```
sp_drop_resource_limit { name, appname }
  [, rangename, limittype, enforced, action,
  scope ]
```

sp_droprowlockpromote
```
sp_droprowlockpromote {"database" | "table"},
  objname
```

sp_dropsegment
```
sp_dropsegment segname, dbname [, device ]
```

sp_dropserver
```
sp_dropserver server [, droplogins ]
```

sp_dropthreshold
```
sp_dropthreshold dbname, segname, free_space
```

sp_drop_time_range
```
sp_drop_time_range name
```

sp_droptype
```
sp_droptype typename
```

sp_dropuser
```
sp_dropuser name_in_db
```

sp_dumpoptimize
```
sp_dumpoptimize [ "archive_space =
  { maximum | minimum | default }" ]
sp_dumpoptimize [ "reserved_threshold =
  { nnn | default }" ]
sp_dumpoptimize [ "allocation_threshold =
  { nnn | default }" ]
```

sp_estspace
```
sp_estspace table_name, no_of_rows [,
  fill_factor [, cols_to_max
  [, textbin_len [, iosec] ] ] ]
```

sp_export_qpgroup
```
sp_export_qpgroup usr, group, tab
```

sp_extendsegment
```
sp_extendsegment segname, dbname, devname
```

sp_extengine
```
sp_extengine 'ejb_server', '{start | stop |
  status}'
```

sp_familylock
```
sp_familylock [fpid1 [, fpid2] ]
```

sp_find_qplan
```
sp_find_qplan pattern [, group ]
```

sp_flushstats
```
sp_flushstats objname
```

sp_forceonline_db
```
sp_forceonline_db dbname,
  { "sa_on" | "sa_off" | "all_users" }
```

sp_forceonline_object
```
sp_forceonline_object dbname, objname, indid,
  { sa_on | sa_off | all_users } [, no_print]
```

sp_forceonline_page
```
sp_forceonline_page dbname, pgid,
  {"sa_on" | "sa_off" | "all_users"}
```

sp_foreignkey
```
sp_foreignkey tabname, pktabname, col1
  [, col2] ... [, col8]
```

sp_freedll
```
sp_freedll dll_name
```

sp_getmessage

```
sp_getmessage message_num, result output
  [, language]
```

sp_grantlogin

Windows NT only

```
sp_grantlogin { login_name | group_name }
  [ "role_list" | default ]
```

sp_ha_admin

```
sp_ha_admin [ cleansessions | help ]
```

sp_help

```
sp_help [objname]
```

sp_helpartition

```
sp_helpartition [table_name]
```

sp_helpcache

```
sp_helpcache { cache_name |
  "cache_size[P|K|M|G]" }
```

sp_helpconfig

```
sp_helpconfig "configname", ["size"]
sp_helpconfig "number of ccbs"
sp_helpconfig "caps per ccb"
sp_helpconfig "average cap size"
```

sp_helpconstraint

```
sp_helpconstraint [objname] [, detail]
```

sp_helpdb

```
sp_helpdb [dbname]
```

sp_helpdevice

```
sp_helpdevice [devname]
```

sp_helpextendedproc

```
sp_helpextendedproc [esp_name]
```

sp_helpexternlogin

Component Integration Services only

```
sp_helpexternlogin [ remote_server
  [, login_name] ]
```

sp_helpgroup

```
sp_helpgroup [grpname]
```

sp_helpindex

```
sp_helpindex objname
```

sp_helpjava

```
sp_helpjava ["class" [, java_class_name
  [, "detail" | "depends" ] ] | "jar"
  [, jar_name [, "depends" ] ] ]
```

sp_helpjoins

```
sp_helpjoins lefttab, righttab
```

sp_helpkey

```
sp_helpkey [tabname]
```

sp_helplanguage
```
sp_helplanguage [language]
```

sp_helplog
```
sp_helplog
```

sp_helpobjectdef
Component Integration Services only
```
sp_helpobjectdef [object_name]
```

sp_help_qpgroup
```
sp_help_qpgroup [ group [, mode ] ]
```

sp_help_qplan
```
sp_help_qplan id [, mode ]
```

sp_helpremotelogin
```
sp_helpremotelogin [remoteserver
  [, remotename] ]
```

sp_help_resource_limit
```
sp_help_resource_limit [name [, appname
  [, limittime [, limitday
  [, scope [, action] ] ] ] ] ]
```

sp_helpprotect
```
sp_helprotect [name [, username [, "grant"
  [, "none"|"granted"|"enabled"|role_name]]]]
```

sp_helpsegment
```
sp_helpsegment [segname]
```

sp_helpserver
```
sp_helpserver [server]
```

sp_helpsort
```
sp_helpsort
```

sp_helptext
```
sp_helptext objname
```

sp_helpthreshold
```
sp_helpthreshold [segname]
```

sp_helpuser
```
sp_helpuser [name_in_db]
```

sp_hidetext
```
sp_hidetext [objname
  [, tabname [, username] ] ]
```

sp_import_qpgroup
```
sp_import_qpgroup tab, usr, group
```

sp_indsuspect
```
sp_indsuspect [tab_name]
```

sp_listsuspect_db
```
sp_listsuspect_db
```

sp_listsuspect_object
```
sp_listsuspect_object [dbname]
```

sp_listsuspect_page
```
sp_listsuspect_page [dbname]
```

sp_lock
```
sp_lock [spid1 [, spid2]]
```

sp_locklogin
```
sp_locklogin [loginame, "{ lock|unlock }"]
```

sp_logdevice
```
sp_logdevice dbname, devname
```

sp_loginconfig
Windows NT only
```
sp_loginconfig ["parameter_name"]
```

sp_logininfo
Windows NT only
```
sp_logininfo [ "login_name" | "group_name" ]
```

sp_logiosize
```
sp_logiosize [ "default" | "size" | "all" ]
```

sp_modifylogin
```
sp_modifylogin account, column, value
```

sp_modify_resource_limit
```
sp_modify_resource_limit {name, appname }
  , rangename , limittype , limitvalue
  , enforced , action , scope
```

sp_modifystats
```
sp_modifystats [database].[owner][table_name],
  column_name,
  REMOVE_SKEW_FROM_DENSITY
or
   sp_modifystats [db].[owner][table_name],
   {column_name} [, column_name] [,...] ] },
   MODIFY_DENSITY,
   {range | total},
   {absolute | total},
   "value"
```

sp_modify_time_range
```
sp_modify_time_range name, startday, endday,
  starttime, endtime
```

sp_modifythreshold
```
sp_modifythreshold dbname, segname, free_space
  [, new_proc_name] [, new_free_space]
  [, new_segname]
```

sp_monitor
```
sp_monitor
```

sp_monitorconfig
```
sp_monitorconfig "configname"
```

sp_object_stats
```
sp_object_stats interval [, top_n
  [, dbname, objname [, rpt_option ] ] ]
```

sp_passthru
Component Integration Services only
```
sp_passthru server, command, errcode, errmsg,
  rowcount
    [, arg1, arg2, ... argn]
```

sp_password
```
sp_password caller_passwd, new_passwd
  [, loginame]
```

sp_placeobject
```
sp_placeobject segname, objname
```

sp_plan_dbccdb
```
sp_plan_dbccdb [dbname]
```

sp_poolconfig
Creating a memory pool in an existing cache, or changing pool size:
```
sp_poolconfig cache_name
  [, "mem_size[P|K|M|G]", "config_poolK"
  [, "affected_poolK"] ]
```
Changing a pool's wash size:
```
sp_poolconfig cache_name, "io_size",
  "wash=size[P|K|M|G]"
```
Changing a pool's asynchronous prefetch percentage:
```
sp_poolconfig cache_name, "io_size",
  "local async prefetch limit=percent"
```

sp_primarykey
```
sp_primarykey tabname, col1
  [, col2, col3, ..., col8]
```

sp_processmail
Windows NT only
```
sp_processmail [subject] [, originator
  [, dbuser [, dbname [, filetype
  [, separator] ] ] ] ]
```

sp_procqmode
```
sp_procqmode [object_name [, detail] ]
```

sp_procxmode
```
sp_procxmode [procname [, tranmode] ]
```

sp_recompile
```
sp_recompile objname
```

sp_remap
```
sp_remap objname
```

sp_remoteoption
```
sp_remoteoption [remoteserver [, loginame
  [, remotename [, optname [, optvalue]]]]]
```

sp_remotesql
Component Integration Services only
```
sp_remotesql server, query
  [, query2, ... , query254]
```

sp_rename
```
sp_rename objname, newname
```

sp_renamedb
```
sp_renamedb dbname, newname
```

sp_rename_qpgroup
```
sp_rename_qpgroup old_name, new_name
```

sp_reportstats
```
sp_reportstats [loginame]
```

sp_revokelogin
Windows NT only
```
sp_revokelogin { login_name | group_name }
```

sp_role
```
sp_role {"grant"|"revoke"}, rolename, loginame
```

sp_sendmsg
```
sp_sendmsg ip_address, port_number, message
```

sp_serveroption
```
sp_serveroption [server, optname, optvalue]
```

sp_setlangalias
```
sp_setlangalias language, alias
```

sp_setpglockpromote
```
sp_setpglockpromote {"database"|"table"},
  objname, new_lwm, new_hwm, new_pct
sp_setpglockpromote server, NULL, new_lwm,
  new_hwm, new_pct
```

sp_setpsexe
```
sp_setpsexe spid, exeattr, value
```

sp_set_qplan
```
sp_set_qplan id, plan
```

sp_setrowlockpromote
```
sp_setrowlockpromote "server", NULL, new_lwm,
  new_hwm, new_pct
sp_setrowlockpromote {"database" | "table"},
  objname, new_lwm, new_hwm, new_pct
```

sp_setsuspect_granularity
```
sp_setsuspect_granularity [dbname
  [, "database" | "page" [, "read_only"] ] ]
```

sp_setsuspect_threshold
```
sp_setsuspect_threshold [dbname [, threshold]]
```

sp_showcontrolinfo
```
sp_showcontrolinfo
  [object_type, object_name, spid]
```

sp_showexeclass
```
sp_showexeclass [execlassname]
```

sp_showplan
```
sp_showplan spid, batch_id output, context_id
  output, stmt_num output
```
Displaying the showplan output for the current SQL statement without specifying the batch_id, context_id, or stmt_num:
```
sp_showplan spid, null, null, null
```

sp_showpsexe
```
sp_showpsexe [spid]
```

sp_spaceused

```
sp_spaceused [objname [,1] ]
```

sp_ssladmin

```
sp_ssladmin { [addcert, certificate_path,
  password]
  [dropcert, certificate_path]
  [lscert]
  [help] }
```

sp_sysmon

```
sp_sysmon begin_sample
sp_sysmon { end_sample | interval }
  [, section [, applmon] ]
sp_sysmon {end_sample|interval} [, applmon]
```

sp_thresholdaction

```
sp_thresholdaction @dbname,
     @segment_name,
     @space_left,
     @status
```

sp_unbindcache

```
sp_unbindcache dbname [, [owner.]tablename
  [, indexname | "text only"] ]
```

sp_unbindcache_all

```
sp_unbindcache_all cache_name
```

sp_unbindefault

```
sp_unbindefault objname [, futureonly]
```

sp_unbindexeclass

```
sp_unbindexeclass object_name, object_type,
  scope
```

sp_unbindmsg

```
sp_unbindmsg constrname
```

sp_unbindrule

```
sp_unbindrule objname [, futureonly]
```

sp_volchanged

```
sp_volchanged session_id, devname, action
  [, fname [, vname] ]
```

sp_who

```
sp_who [loginame | "spid"]
```

Catalog Stored Procedures

sp_column_privileges
```
sp_column_privileges table_name [, table_owner
  [, table_qualifier [, column_name] ] ]
```

sp_columns
```
sp_columns table_name [, table_owner ]
  [, table_qualifier] [, column_name]
```

sp_databases
```
sp_databases
```

sp_datatype_info
```
sp_datatype_info [data_type]
```

sp_fkeys
```
sp_fkeys pktable_name [, pktable_owner]
  [, pktable_qualifier] [, fktable_name]
  [, fktable_owner] [, fktable_qualifier]
```

sp_pkeys
```
sp_pkeys table_name [, table_owner]
  [, table_qualifier]
```

sp_server_info
```
sp_server_info [attribute_id]
```

sp_special_columns
```
sp_special_columns table_name [, table_owner]
  [, table_qualifier] [, col_type]
```

sp_sproc_columns
```
sp_sproc_columns procedure_name
  [, procedure_owner] [, procedure_qualifier]
  [, column_name]
```

sp_statistics
```
sp_statistics table_name [, table_owner]
  [, table_qualifier] [, index_name]
  [, is_unique]
```

sp_stored_procedures
```
sp_stored_procedures [sp_name [, sp_owner
  [, sp_qualifier] ] ]
```

sp_table_privileges
```
sp_table_privileges table_name [, table_owner
  [, table_qualifier] ]
```

sp_tables
```
sp_tables [table_name] [, table_owner]
  [, table_qualifier] [, table_type]
```

System Extended Stored Procedures

xp_cmdshell

```
xp_cmdshell command [, no_output]
```

xp_deletemail

Windows NT only

```
xp_deletemail [msg_id]
```

xp_enumgroups

Windows NT only

```
xp_enumgroups [domain_name]
```

xp_findnextmsg

Windows NT only

```
xp_findnextmsg @msg_id = @msg_id output [,
  type] [, unread_only = { true|false }]
```

xp_logevent

Windows NT only

```
xp_logevent error_number, message [, type]
```

xp_readmail

Windows NT only

```
xp_readmail [msg_id][, recipients output]
   [, sender output]
   [, date_received output]
   [, subject output]
   [, cc output]
   [, message output]
   [, attachments output]
   [, suppress_attach = { true | false } ]
   [, peek = { true | false } ]
   [, unread = { true | false } ]
   [, msg_length output]
   [, bytes_to_skip [output] ]
   [, type [output] ]
```

xp_sendmail

Windows NT only

```
xp_sendmail recipient [; recipient]...
   [, subject]
   [, cc_recipient]...
   [, bcc_recipient]...
   [, { query | message } ]
   [, attachname]
   [, attach_result = { true | false } ]
   [, echo_error = { true | false } ]
   [, include_file [, include_file]...]
   [, no_column_header = { true | false } ]
   [, no_output = { true | false } ]
   [, width]
   [, separator]
   [, dbuser]
   [, dbname]
   [, type]
   [, include_query = { true | false } ]
```

xp_startmail

Windows NT only

```
xp_startmail [mail_user] [, mail_password]
```

xp_stopmail

Windows NT only

```
xp_stopmail
```

dbcc Stored Procedures

sp_dbcc_alterws
```
sp_dbcc_alterws dbname, wsname, "wssize[K|M]"
```

sp_dbcc_configreport
```
sp_dbcc_configreport [dbname]
```

sp_dbcc_createws
```
sp_dbcc_createws dbname, segname, [wsname],
  wstype, "wssize[ K | M ]"
```

sp_dbcc_deletedb
```
sp_dbcc_deletedb [dbname]
```

sp_dbcc_deletehistory
```
sp_dbcc_deletehistory [cutoffdate [, dbname] ]
```

sp_dbcc_differentialreport
```
sp_dbcc_differentialreport [dbname
  [, objectname]], [db_op]
  [, "date1" [, "date2"]]
```

sp_dbcc_evaluatedb
```
sp_dbcc_evaluatedb [dbname]
```

sp_dbcc_faultreport
```
sp_dbcc_faultreport [report_type [, dbname
  [, objectname [, date ] ] ] ]
```

sp_dbcc_fullreport
```
sp_dbcc_fullreport [dbname [, objectname
  [, date] ] ]
```

sp_dbcc_runcheck
```
sp_dbcc_runcheck dbname [, user_proc]
```

sp_dbcc_statisticsreport
```
sp_dbcc_statisticsreport [dbname [, objectname
  [, date] ] ]
```

sp_dbcc_summaryreport
```
sp_dbcc_summaryreport [dbname [, op_name] ]
```

sp_dbcc_updateconfig
```
sp_dbcc_updateconfig dbname, type, "str1"
  [, "str2"]
```

Utilities

backupserver

bcksrvr.exe in Windows NT

```
backupserver [ -A ] [ -D ] [ -m ] [ -p ] [ -s ]
  [ -ctape_config_file ]
  [ -Cserver_connections ]
  [ -eerror_log_file ]
  [ -Iinterfaces_file ]
  [ -JSybase_character_set_name ]
  [ -LSybase_language_name ]
  [ -Msybmultbuf_binary ]
  [ -Nnetwork_connections ]
  [ -Pactive_service_threads ]
  [ -Sb_servername ]
  [ -Ttrace_value ]
  [ -Vversion_number ]

or
   backupserver -v
```

bcp

Also use for bcp_r, bcp_dce

```
bcp [ [dbame.]owner.]table_name[:slice_number]
  {in | out} datafile
  [ -labeled ]
  [ -c ] [ -E ] [ -n ] [ -N ] [ -Q ] [ -X ]
  [ -a display_charset ]
  [ -A packet size ]
  [ -b batchsize ]
  [ -e errfile ]
  [ -f formatfile ]
  [ -F firstrow ]
  [ -g id_start_value ]
  [ -I interfaces_file ]
  [ -J client character set ]
  [ -K keytab_file ]
  [ -L lastrow ]
  [ -m maxerrors ]
  [ -M LabelName LabelValue ]
  [ -P password ]
  [ -r row_terminator ]
  [ -R remote_server_principal ]
  [ -q datafile_charset ]
  [ -S server ]
  [ -t field_terminator ]
  [ -T text_or_image_size ]
  -U username
  [ -V [security_options] ]
  [ -z language ]
  [ -Z security_mechanism ]

or
   bcp -v
```

Note: [-N] is for Windows NT only

certauth

```
certauth [-r]
  [-C caCert_file]
  [-Q request_filename]
  [-K caKey_filename]
  [-O SignedCert_filename]
  [-P caPassword]
  [-T valid_time]

or
   certauth -v
```

certpk12

```
certpk12 { -O Pkcs12_file | -I Pkcs12_file }
  [ -C Cert_file ]
  [ -E Pkcs12_password ]
  [ -K Key_file ]
  [ -P key_password ]

or
   certpk12 -v
```

certreq

```
certreq
  [ -F input_file ]
  [ -K PK_filename ]
  [ -P password ]
  [ -R request_filename ]

or
   certreq -v
```

charset

```
charset
  [ -I interface ]
  [ -P password ]
  [ -S server ]
  sort_order
  [ charset ]

or
   charset -v
```

cobpre

Also use for cobpre_r and copbre_dce.

```
cobpre [ -a ] [ -b ] [ -c ] [ -d ] [ -e ]
  [ -f ] [ -h ] [ -m ] [ -p ] [ -q ] [ -r ]
  [ -v ] [ -w ] [ -x ]
  [ -B compatibility_mode ]
  [ -C compiler ]
  [ -D database ]
  [ -F fips_level ]
  [ -G [isql_file_name] ]
  [ -I include_path_name ]
  [ -J charset_locale_name ]
  [ -K syntax_level ]
  [ -L [listing_file_name] ]
  [ -N interfacefile_name ]
  [ -O target_file_name ]
  [ -P password ]
  [ -S server_name ]
  [ -T tag ]
  [ -U userid ]
  [ -V version_number ]
  [ -X application_name ]
  [ -Z language_locale_name ]
  [ @options_file ]
  filename[.ext] ...
```

cpre

Also use for cpre_r and cpre_dce.

```
cpre [ -a ] [ -b ] [ -c ] [ -d ] [ -f ] [ -h ]
  [ -l ] [ -m ] [ -p ] [ -r ] [ -s ] [ -v ]
  [ -w ] [ -x ]
  [ -B compatibility_mode ]
  [ -C compiler ]
  [ -D database ]
  [ -F fips_level ]
  [ -G [isql_file_name] ]
  [ -I include_path_name ]
  [ -J charset_locale_name ]
  [ -K syntax_level ]
  [ -L [listing_file_name] ]
  [ -N interfacefile_name ]
  [ -O target_file_name ]
```

```
   [ -P password ]
   [ -S server_name ]
   [ -T tag ]
   [ -U userid ]
   [ -V version_number ]
   [ -X application_name ]
   [ -Z language_locale_name ]
   [ @options_file ]
   filename[.ext]
```

dataserver

UNIX platforms only

```
dataserver [ -f ] [ -g ] [ -G ] [ -h ] [ -H ]
   [ -m ] [ -P ] [ -q ] [ -v ] [ -X ]
   [ -a path_to_CAPs_directive_file ]
   [ -b master_device_size [k|K|m|M|g|G]]
   [ -c config_file_for_server ]
   [ -d device_name ]
   [ -e path_to_error_log ]
   [ -i interfaces_file_directory ]
   [ -r mastermirror ]
   [ -K keytab_file ]
   [ -L config_file_name_for_connectivity]
   [ -M shared_memory_repository_directory ]
   [ -p sa_login_name ]
   [ -r mirror disk name ]
   [ -s servername ]
   [ -T trace flag ]
   [ -u { login_name | role_name }
   [ -w master | model database ]
   [ -y [password] ]
   [ -z page size [ k | K ] ]
```

dataxtr

```
dataxtr
```

ddlgen

```
ddlgen
  -Uuser_name
  -Ppassword
  -Shost_name:port_number
  [ -Tobject_type ]
```

For a list of valid object types, see Table 6-3 in the *Utility Guide*.

```
   [ -Nobject_name ]
   [ -Ddb_name ]
   [ -Xextended_object_type ]
   [ -Ooutput_file_name ]
   [ -Eerror_log_file ]
```

or

```
   ddlgen -v
```

defncopy

Also use for defncopy_r, defncopy_dce

UNIX platforms

```
defncopy [ -X ]
   [ -a display_charset ]
   [ -I interfaces_file ]
   [ -J [client_charset] ]
   [ -K key_tab ]
   [ -P password ]
   [ -R remote_server_principal ]
   [ -S [server] ]
   [ -U username ]
   [ -V security_options ]
   [ -z language ]
   [ -Z security_mechanism ]
   { in filename dbname | out filename dbname
     [owner.]objectname [[owner.]objectname...]}
```

```
or
  defncopy -v
```
Windows NT:
```
defncopy
  [ -X ]
  [ -a display_charset ]
  [ -I sqlini_file ]
  [ -J [client_charset] ]
  [ -P password ]
  [ -S server ]
  [ -U username ]
  [ -z language ]
  { in filename dbname | out filename dbname
    [owner.]objectname [ [owner.]objectname]...}
or
  defncopy -v
```

dscp

Also use for dscp_r, dscp_dce
```
dscp [ -p ]
or
  dscp -v
```

dsedit

Also use for dsedit_r, dsedit_dce
```
dsedit
or
  dsedit -v
```

extractjava

extrjava.exe in Windows NT
```
extractjava
  -j jar_name
  -f file_name
  [ -h ] [ -v ]
  [ -a display_charset ]
  [ -D database_name ]
  [ -I interfaces_file ]
  [ -J client_charset ]
  [ -P password ]
  [ -S server_name ]
  [ -t timeout ]
  [ -U user_name ]
  [ -z language ]
```

installjava

instjava.exe in Windows NT
```
installjava
  -f file_name
  [ -new | -update ]
  [ -h ] [ -v ]
  [ -a display_charset ]
  [ -D database_name ]
  [ -I interfaces_file ]
  [ -j jar_name ]
  [ -J client_charset ]
  [ -P password ]
  [ -S server_name ]
  [ -t timeout ]
  [ -U user_name ]
  [ -z language ]
```

isql

Also use for isql_r, isql_dce
```
isql [ -b ] [ -e ] [ -F ] [ -p ] [ -n ] [ -X ]
  [ -Y ] [ -Q ]
  [ -a display_charset ]
  [ -A packet_size ]
  [ -c cmdend ]
```

```
  [ -D database ]
  [ -E editor ]
  [ -h header ]
  [ -H hostname ]
  [ -i inputfile ]
  [ -I interfaces_file ]
  [ -J client_charset ]
  [ -K keytab_file ] (UNIX only)
  [ -l login_timeout ]
  [ -m errorlevel ]
  [ -M labelname labelvalue ] (UNIX only)
  [ -o outputfile ]
  [ -P password ]
  [ -R remote_server_principal ]
  [ -s col_separator ]
  [ -S server_name ]
  [ -t timeout ]
  -U username
  [ -V [security_options] ] (UNIX only)
  [ -w column_width ]
  [ -z localename ]
  [ -Z security_mechanism ]
```

or

```
  isql -v
```

To terminate a command: go
To clear the query buffer: reset
To call the default editor: vi
To execute an operating system command: !! command
To exit from isql: quit or exit

langinstall

langinst.exe in Windows NT

```
langinstall
  [ -I path_to_interfaces_file]
  [ -P password ]
  [ -R release_number ]
  [ -S server ]
  [ -U user ]
  language
  charset
```

or

```
  langinstall -v
```

optdiag

```
optdiag [ binary ] [ simulate ] statistics
  { -i input_file |
    database[.owner[.[table[.column ] ] ] ]
      [ -o output_file ] }
  [ -h ] [ -s ] [ -v ]
  [ -a display_charset ]
  [ -I interfaces_file ]
  [ -J client_character_set ]
  [ -P password ]
  [ -S server ]
  [ -T trace_value ]
  [ -U username ]
  [ -z language ]
```

pwdcrypt

```
pwdcrypt
```

showserver

```
showserver
```

sqldbgr

```
sqldbgr
  -U username
  -P password
  -S host_name:port_number
```

sqlloc

```
sqlloc
  [ -I Interfaces_file ]
  [ -P Password ]
  [ -r Resource_file ]
  [ -s Sybase_Dir ]
  [ -S Server ]
  [ -U User ]

or
  sqlloc -v
```

sqllocres

```
sqllocres
  [ -I Interfaces_file ]
  [ -P Password ]
  [ -r Resource_file ]
  [ -s Sybase_Dir ]
  [ -S Server ]
  [ -U User ]

or
  sqlloc -v
```

sqlsrvr

Windows NT only

```
sqlsrvr
  [ -f ] [ -g ] [ -G ] [ -h ] [ -H ] [ -m ]
  [ -P ] [ -q ] [ -v ] [ -X ]
  [ -a path to CAPS directive file ]
  [ -b master_device_size ]
  [ -c configuration_file_for_server ]
  [ -d device_name ]
  [ -e path_to_error_log ]
  [ -i sql.ini_file_directory ]
  [ -K keytab_file ]
  [ -L config_file_name_for_connectivity ]
  [ -M shared_memory_repository_directory ]
  [ -p sa_login_name ]
  [ -r mirror_disk_name ]
  [ -s server_name ]
  [ -T trace_flag ]
  [ -u sa/sso_name ]
  [ -w master | model_database ]
  [ -y [password] ]
  [ -z page_size ]
```

sqlupgrade

```
sqlupgrade
  [ -r Resource_File ]
  [ -s Sybase_Dir ]

or
   sqlupgrade -v
```

sqlupgraderes

```
sqlupgraderes
  [ -r Resource_File ]
  [ -s Sybase_Dir ]

or
   sqllocres -v
```

srvbuild

```
srvbuild
  [ -I interfaces_file ]
  [ -r resource_file ]
  [ -s sybase_dir ]

or
   srvbuild -v
```

srvbuildres

```
srvbuildres
  [ -I interfaces_file ]
  [ -r resource_file ]
  [ -s sybase_dir ]
or
   srvbuildres -v
```

startserver

```
startserver
  [ [ -f runserverfile ] [ -m ] ] ...
```

xpserver

```
xpserver
   -SXP_Server
  [ -u ] [ -v ] [ -x ]
  [ -Iinterfaces_file ]
  [ -ppriority ]
  [ -sstack_size ]
```