

Open a Channel

OPEN(*channel*, *mode:submode*, *file*, ● **ALLOC:blocks**, ● **LOCK:spec**, ● **OPTIONS:"runtime_options"**, ● **POSITION:spec**,
 & ● **RECSIZ:size**, ● **SHARE:spec**, ● **TEMPFILE:spec**, ● **FDL:string**, ● **GUIWIND**, ● **BKTSIZ:size**, ● **BLKSIZ:size**,
 & ● **BUFNUM:number**, ● **BUFSIZ:blocks**, ● **CONTIG:value**, ● **DEQ:blocks**, ● **RECTYPE:type**) [*error_List*]

● **ALLOC:blocks** Pre-allocate number of blocks

blocks Number of 512-byte blocks to allocate (n)

● **BKTSIZ:size** Set bucket size

size RMS bucket size (n)

● **BLKSIZ:size** Set block size

size RMS block size (n)

● **BUFNUM:number** Set number of process local I/O buffers

number Number of process local I/O buffers (0-127) (n)

● **BUFSIZ:blocks** Set buffer size

blocks Number of blocks to allocate (1-127) (n)

● **CONTIG:value** Allocate file contiguity

0 Not contiguously
 1 Contiguously
 2 Best-try basis

● **DEQ:blocks** Set number of blocks to add

blocks Number of blocks to add (n)

● **FDL:string** Specify FDL or XDL string or file specification

string FDL or XDL string or file spec (a)

● **GUIWIND** Open new application window

● **LOCK:spec** Control record locking

0 or Q_NO_LOCK No locking
 1 or Q_AUTO_LOCK Automatic locking
 2 or Q_MANUAL_LOCK Manual locking
 3 or Q_NO_TLOCK No locking or probing of index tree

● **OPTIONS:"runtime_options, ..."** Specify runtime options

runtime_options List of options separated by commas

● **POSITION:spec** Define file position

-1 or Q_IGNPOS Ignore POSITION qualifier
 0 or Q_FIRST First record in file
 1 or Q_LAST Last record in file
 2 or Q_EOF End of file without establishing current record
 3 or Q_BOF Beginning of file without establishing current record

● **RECSIZ:size** Specify record size for file

n Set record size to n
 0 Ignore RECSIZ qualifier
 -1 Set record size to length of first record

● **RECTYPE:type** Define type of record to write to file

0 Undefined format
 1 Fixed-length
 2 Variable-length
 3 VFC (variable-fixed control)
 4 Stream
 5 Stream format with line-feed carriage control
 6 Stream format with carriage-return carriage control

● **SHARE:spec** Control file access

0 or Q_EXCL_RW No access for other users
 1 or Q_EXCL_RO Read-only access for other users
 2 or Q_NO_EXC Read/write access for other users

● **TEMPFILE:spec** Define intermediate file for output mode

temp_spec Intermediate output file specification

Send Output to a Device or File



DISPLAY(*channel*, *item*, ...)

\$SCR_ATT(*function*, ...) Modify screen attributes

CLEAR All attributes off
 BOLD Bolding on
 UNDER Underlining on
 BLINK Blinking on
 REVERSE Reverse video on
 GON Graphics on
 GOFF Graphics off
 SAVE Save attributes (Unix, OpenVMS)
 RESTORE Restore attributes from last SAVE (Unix, OpenVMS)

\$SRC_POS(*row*, *col*) Reset absolute position of cursor

\$SCR_MOV(*row_change*, *col_change*) Move cursor relative to current position

\$SCR_CLR(*function*) Clear specific portion of screen

SCREEN Entire screen
 EOL Current position to end of line
 EOS Current position to end of screen
 LINE Current line
 BOL Beginning of line to current position
 BOS Beginning of screen to current position

I/O Statements

ACCEPT(*channel, variable, Label, ● WAIT:wait*)
Receive character from channel

CLOSE *channel, ...*
Close channel

DELETE(*channel*)
Delete record from ISAM file

DISPLAY(*channel, item, ...*)
Send output to device or file

FIND(*channel, record, key_spec, ● GETRFA:newrfa, & ● KEYNUM:spec, ● LOCK:spec, ● MATCH:spec, & ● POSITION:spec, ● RFA:match, ● WAIT:spec*)
Find record

FORMS(*channel, control_code*)
Process ASCII control codes

GET(*channel, data_area, record*)
Receive data from channel

GETS(*channel, data_area, Label, ● WAIT:spec, & ● MASK:act_char*)
Get sequential binary data

PUT(*channel, data_area, record*)
Write fixed-length data

PUS(*channel, data_area, Label*)
Write sequential fixed-length data

READ(*channel, data_area, key_spec, ● GETRFA:new, & ● KEYNUM:spec, ● LOCK:spec, ● MATCH:spec, & ● NOFILL, ● POSITION:spec, ● RFA:match, & ● WAIT:spec*)
Read specific record

READS(*channel, data_area, Label, ● REVERSE, & ● DIRECTION:spec, ● GETRFA:new, ● LOCK:spec, & ● NOFILL, ● WAIT:spec*)
Read next sequential record

STORE(*channel, data_area, ● GETRFA:new, & ● LOCK:spec*)
Store record to ISAM file

UNLOCK *channel, ● RFA:match_rfa*
Release record lock on channel

WRITE(*channel, data_area, record, & ● POSITION:spec, ● GETRFA:new, ● RFA:match*)
Write record to file

WRITES(*channel, data_area, Label, ● GETRFA:new*)
Write next sequential record

I/O Statement Qualifiers

● DIRECTION:spec	Set direction of READS
0 or Q_IGNDIR	Ignore DIRECTION qualifier
1 or Q_FORWARD	Read next sequential record in forward direction
2 or Q_REVERSE	Read next sequential record in reverse direction
● GETRFA:new	Return record's RFA
rfa	Returned RFA (a6)
grfa	Returned global RFA (GRFA) (a10)
● KEYNUM:spec	Specify key of reference
0 or Q_PRIMARY	Primary key
1 or Q_ALT1	First alternative key
...	
7 or Q_ALT7	Seventh alternative key
● LOCK:spec	Control record Locking
0 or Q_NO_LOCK	No locking
1 or Q_AUTO_LOCK	Automatic locking
2 or Q_MANUAL_LOCK	Manual locking
3 or Q_NO_TLOCK	No locking or probing of index tree
● MASK:act_char	Specify activation characters
act_char	Array of integers with total size of 32 bytes that must start on aligned i4 boundary (i)
● MATCH:spec	Define how record is located
0 or Q_GEQ	Greater or equal to key spec
1 or Q_EQ	Equal to key spec
2 or Q_GTR	Greater than key spec
3 or Q_RFA	RFA specified by RFA qualifier
4 or Q_GEN	Match key value equal or next in sequence to key spec
5 or Q_SEQ	Next sequential record following current record; ignore key spec
● NOFILL	Prevent data area from being padded with blanks
● POSITION:spec	Define file position
-1 or Q_IGNPOS	Ignore POSITION qualifier
0 or Q_FIRST	First record in file
1 or Q_LAST	Last record in file
2 or Q_EOF	End of file without establishing current record
3 or Q_BOF	Beginning of file without establishing current record
● REVERSE	Read sequentially in reverse direction
● RFA:match	Locate record with specified RFA
rfa	RFA (a6)
grfa	Global RFA (GRFA) (a10)
● WAIT:spec	Specify wait time for I/O statement to finish processing
0 or Q_NOWAIT	Don't wait if I/O processing can't complete
1 or Q_WAIT	Wait until I/O processing completes
n	Wait up to n seconds for I/O processing to complete



Resource Center: synergex.com/welcomeRC
Documentation: synergex.com/docs
Email: support@synergex.com
Call: 800.366.3472
916.635.7300