Variables & Job Exit Codes

Here you can find the compendium of Slurm environment variables and exit codes for a quick reference.

1.0 INPUT ENVIRONMENT VARIABLES

Upon startup, sbatch will read and handle the options set in the following environment variables. Note that environment variables will override any options set in a batch script, and command line options will override any environment variables.

Variable Name	Equals
SBATCH_ACCOUNT	Same as -A, --account
SBATCH_ACCTG_FREQ	Same as --acctg-freq
SBATCH_ARRAY_INX	Same as -a, --array
SBATCH_BLRTS_IMAGE	Same as --blrts-image
SBATCH_CHECKPOINT	Same as --checkpoint
SBATCH_CHECKPOINT_DIR	Same as --checkpoint-dir
SBATCH_CLUSTERS or SLURM_CLUSTERS	Same as --clusters
SBATCH_CNLOAD_IMAGE	Same as --cnload-image
SBATCH_CONN_TYPE	Same as --conn-type
SBATCH_CONSTRAINT	Same as -C, --constraint
SBATCH_CORE_SPEC	Same as --core-spec
SBATCH_DEBUG	Same as -v, --verbose
SBATCH_DELAY_BOOT	Same as --delay-boot
SBATCH_DISTRIBUTION	Same as -m, --distribution
SBATCH_EXCLUSIVE	Same as --exclusive
SBATCH_EXPORT	Same as --export
SBATCH_GEOMETRY	Same as -g, --geometry
SBATCH_GET_USER_ENV	Same as --get-user-env
SBATCH_GRES_FLAGS	Same as --gres-flags
SBATCH_HINT or SLURM_HINT	Same as --hint
SBATCH_IGNORE_PBS	Same as --ignore-pbs

SBATCH_IMMEDIATE	Same as -I, --immediate
SBATCH_IOLOAD_IMAGE	Same as --ioload-image
SBATCH_JOBID	Same as --jobid
SBATCH_JOB_NAME	Same as -J, --job-name
SBATCH_LINUX_IMAGE	Same as --linux-image
SBATCH_MEM_BIND	Same as --mem-bind
SBATCH_MLOADER_IMAGE	Same as --mloader-image
SBATCH_NETWORK	Same as --network
SBATCH_NO_REQUEUE	Same as --no-requeue
SBATCH_NO_ROTATE	Same as -R, --no-rotate
SBATCH_OPEN_MODE	Same as --open-mode
SBATCH_OVERCOMMIT	Same as -O, --overcommit
SBATCH_PARTITION	Same as -p, --partition
SBATCH_POWER	Same as --power
SBATCH_PROFILE	Same as --profile
SBATCH_QOS	Same as --qos
SBATCH_RAMDISK_IMAGE	Same as --ramdisk-image
SBATCH_RESERVATION	Same as --reservation
SBATCH_REQ_SWITCH	When a tree topology is used, this defines the maximum count of switches desired for the job allocation and optionally the maximum time to wait for that number of switches. See --switches
SBATCH_REQUEUE	Same as --requeue
SBATCH_SIGNAL	Same as --signal
SBATCH_SPREAD_JOB	Same as --spread-job
SBATCH_THREAD_SPEC	Same as --thread-spec
SBATCH_TIMELIMIT	Same as -t, --time
SBATCH_USE_MIN_NODES	Same as --use-min-nodes
SBATCH_WAIT	Same as -W, --wait
SBATCH_WAIT_ALL_NODES	Same as --wait-all-nodes
SBATCH_WAIT4SWITCH	Max time waiting for requested switches. See --switches
SBATCH_WCKEY	Same as --wckey

SLURM_CONF	The location of the Slurm configuration file.
SLURM_EXIT_ERROR	Specifies the exit code generated when a Slurm error occurs (e.g. invalid options). This can be used by a script to distinguish application exit codes from various Slurm error conditions.
SLURM_STEP_KILLED_MSG_NODE_ID=ID	If set, only the specified node will log when the job or step are killed by a signal.

2.0 OUTPUT ENVIRONMENT VARIABLES

The Slurm controller will set the following variables in the environment of the batch script.

Variable Name	**Equals**
BASIL_RESERVATION_ID	The reservation ID on Cray systems running ALPS/BASIL only.
MPIRUN_NOALLOCATE	Do not allocate a block on Blue Gene L/P systems only.
MPIRUN_NOFREE	Do not free a block on Blue Gene L/P systems only.
MPIRUN_PARTITION	The block name on Blue Gene systems only.
SBATCH_MEM_BIND	Set to value of the --mem-bind option.
SBATCH_MEM_BIND_LIST	Set to bit mask used for memory binding.
SBATCH_MEM_BIND_PREFER	Set to "prefer" if the --mem-bind option includes the prefer option.
SBATCH_MEM_BIND_TYPE	Set to the memory binding type specified with the --mem-bind option. Possible values are "none", "rank", "map_map", "mask_mem" and "local".
SBATCH_MEM_BIND_VERBOSE	Set to "verbose" if the --mem-bind option includes the verbose option. Set to "quiet" otherwise.
SLURM_*_PACK_GROUP_#	For a heterogenous job allocation, the environment variables are set separately for each component.
SLURM_ARRAY_TASK_COUNT	Total number of tasks in a job array.
SLURM_ARRAY_TASK_ID	Job array ID (index) number.
SLURM_ARRAY_TASK_MAX	Job array's maximum ID (index) number.
SLURM_ARRAY_TASK_MIN	Job array's minimum ID (index) number.
SLURM_ARRAY_TASK_STEP	Job array's index step size.
SLURM_ARRAY_JOB_ID	Job array's master job ID number.
SLURM_CHECKPOINT_IMAGE_DIR	Directory into which checkpoint images should be written if specified on the execute line.
SLURM_CLUSTER_NAME	Name of the cluster on which the job is executing.

SLURM_CPUS_ON_NODE	Number of CPUS on the allocated node.
SLURM_CPUS_PER_TASK	Number of cpus requested per task. Only set if the --cpus-per-task option is specified.
SLURM_DISTRIBUTION	Same as -m, --distribution
SLURM_GTIDS	Global task IDs running on this node. Zero origin and comma separated.
SLURM_JOB_ACCOUNT	Account name associated of the job allocation.
SLURM_JOB_ID (and SLURM_JOBID for backwards compatibility)	The ID of the job allocation.
SLURM_JOB_CPUS_PER_NODE	Count of processors available to the job on this node. Note the select/linear plugin allocates entire nodes to jobs, so the value indicates the total count of CPUs on the node. The select/cons_res plugin allocates individual processors to jobs, so this number indicates the number of processors on this node allocated to the job.
SLURM_JOB_DEPENDENCY	Set to value of the --dependency option.
SLURM_JOB_NAME	Name of the job.
SLURM_JOB_NODELIST (and SLURM_NODELIST for backwards compatibility)	List of nodes allocated to the job.
SLURM_JOB_NUM_NODES (and SLURM_NNODES for backwards compatibility)	Total number of nodes in the job's resource allocation.
SLURM_JOB_PARTITION	Name of the partition in which the job is running.
SLURM_JOB_QOS	Quality Of Service (QOS) of the job allocation.
SLURM_JOB_RESERVATION	Advanced reservation containing the job allocation, if any.
SLURM_LOCALID	Node local task ID for the process within a job.
SLURM_MEM_PER_CPU	Same as --mem-per-cpu
SLURM_MEM_PER_NODE	Same as --mem
SLURM_NODE_ALIASES	Sets of node name, communication address and hostname for nodes allocated to the job from the cloud. Each element in the set if colon separated and each set is comma separated. For example: SLURM_NODE_ALIASES=ec0:1.2.3.4:foo,ec1:1.2.3.5:bar
SLURM_NODEID	ID of the nodes allocated.

SLURM_NTASKS (and SLURM_NPROCS for backwards compatibility)	Same as -n, --ntasks
SLURM_NTASKS_PER_CORE	Number of tasks requested per core. Only set if the --ntasks-per-core option is specified.
SLURM_NTASKS_PER_NODE	Number of tasks requested per node. Only set if the --ntasks-per-node option is specified.
SLURM_NTASKS_PER_SOCKET	Number of tasks requested per socket. Only set if the --ntasks-per-socket option is specified.
SLURM_PACK_SIZE	Set to count of components in heterogeneous job.
SLURM_PRIO_PROCESS	The scheduling priority (nice value) at the time of job submission. This value is propagated to the spawned processes.
SLURM_PROCID	The MPI rank (or relative process ID) of the current process
SLURM_PROFILE	Same as --profile
SLURM_RESTART_COUNT	If the job has been restarted due to system failure or has been explicitly requeued, this will be sent to the number of times the job has been restarted.
SLURM_SUBMIT_DIR	The directory from which sbatch was invoked.
SLURM_SUBMIT_HOST	The hostname of the computer from which sbatch was invoked.
SLURM_TASKS_PER_NODE	Number of tasks to be initiated on each node. Values are comma separated and in the same order as SLURM_JOB_NODELIST. If two or more consecutive nodes are to have the same task count, that count is followed by "(x#)" where "#" is the repetition count. For example, "SLURM_TASKS_PER_NODE=2(x3),1" indicates that the first three nodes will each execute three tasks and the fourth node will execute one task.
SLURM_TASK_PID	The process ID of the task being started.
SLURM_TOPOLOGY_ADDR	This is set only if the system has the topology/tree plugin configured. The value will be set to the names network switches which may be involved in the job's communications from the system's top level switch down to the leaf switch and ending with node name. A period is used to separate each hardware component name.
SLURM_TOPOLOGY_ADDR_PATTERN	This is set only if the system has the topology/tree plugin configured. The value will be set component types listed in SLURM_TOPOLOGY_ADDR. Each component will be identified as either "switch" or "node". A period is used to separate each hardware component type.
SLURMD_NODENAME	Name of the node running the job script.

2.1 Filename patterns

sbatch allows for a filename pattern to contain one or more replacement symbols, which are a percent sign "%" followed by a letter (e.g. %j).

```
%% The character "%".

%A Job array's master job allocation number.

%a Job array ID (index) number.

%J jobid.stepid of the running job. (e.g. "128.0")

%j jobid of the running job.

%N short hostname. This will create a separate IO file per node.

%n Node identifier relative to current job (e.g. "0" is the first node
of the running job) This will create a separate IO file per node.

%s stepid of the running job.

%t task identifier (rank) relative to current job. This will create a
separate IO file per task.

%u User name.

%x Job name.
```

Some examples of how the format string may be used for a 4 task job step with a Job ID of 128 and step id of 0 are included below:

```
job%J.out
```
job128.0.out

```
job%4j.out
job0128.out

job%j-%2t.out
job128-00.out, job128-01.out, ...
```

3.0 JOB EXIT CODES

The exit code from a batch job is a standard Unix termination signal and exit code 0 means successful completion. Codes 1-127 are generated from the job calling exit() with a non-zero value to indicate an error. Codes 129-255 represent jobs terminated by Unix signals.

Signal Name	Signal Number	Exit Type	Reason
SIGHUP	1	Term	Hangup detected on controlling terminal or death of controlling process
SIGINT	2	Term	Interrupt from keyboard
SIGQUIT	3	Core	Quit from keyboard
SIGILL	4	Core	Illegal Instruction
SIGABRT	6	Core	Abort signal from abort(3)
SIGFPE	8	Core	Floating point exception
SIGKILL	9	Term	Kill signal
SIGSEGV	11	Core	Invalid memory reference
SIGPIPE	13	Term	Broken pipe: write to pipe with no readers
SIGALRM	14	Term	Timer signal from alarm(2)
SIGTERM	15	Term	Termination signal

Exit Code	Reason
9	CPU time limit.
64	Your job was running out of CPU time. Allocate more resources, eg. CPU time limit.
125	An ErrMsg(severe) was reached.
127	System has a problem(?), contact administrators.

130	Run out of CPU or swap time. If suspecting swap time, check for memory leaks.
131	Run out of CPU or swap time. If suspecting swap time, check for memory leaks.
134	The job killed with an abort signal, and you probably got core dumped. Possible causes: assert() or an ErrMsg(fatal) hit. Possible run-time bug. Use a debugger to find out what's wrong.
137	The job was killed because it exceeded the time limit.
139	Segmentation violation. Usually indicates a pointer error.
140	The job exceeded the "wall clock" time limit (as opposed to the CPU time limit).