Name	Version	Description
agent.hostname	1.3	currently, same as system.hostname
agent.ping	1.0	Ping to the server (IP)
agent.version	1.0	Version of zabbix_agent
kernel.numlock	1.3	Current number of locks
kernel.maxproc	1.0	Maximum number of processes on the server (ever)
logical.exists[LOGNAME,TABLENAME]	1.3	Check if LOGNAME exists in table TABLENAME default LNM$FILE_DEV
logical.value[LOGNAME,TABLENAME]	1.3	Value of LOGNAME in table TABLENAME default LNM$FILE_DEV return ‘’ is LOGNAME doesn’t exist
net.service.exists(HOST,PORT)	1.3	Check if the host HOST on port PORT is accessible. Return 1 if ok, 0 if not
system.hostname	1.0	hostname
system.uname	1.0	Host information
system.boottime	1.0	Boot time
system.uptime	1.0	Host Uptime (in seconds)
system.localtime	1.0	Host local time
system.bufio	1.0	Host BUFIO
system.dirio	1.0	Host DIRIO
system.cpu.load[,AVG]	1.0	Total CPU load (AVG in (avg1, avg5, avg15)
system.mem[memsize]	1.2	Memory size (in KB)
system.mem[page_size]	1.2	Page size (in Bytes)
system.mem[total_pages]	1.2	Total number of pages on the system
system.mem[contig_gblpages]	1.3	Current number of contiguous global pages
system.mem[free_gblpages]	1.2	Free global pages
system.mem[used_gblpagcnt]	1.2	used global pages
system.mem[used_gblpagmax]	1.2	Max number of used global pages
system.mem[free_gblsects]	1.2	Free global sections
system.mem[pagefile_page]	1.2	Number of pages in the page files
system.mem[pagefile_free]	1.2	Free pages in the page files
system.mem[swapfile_page]	1.2	Number of pages in the swap files
system.mem[swapfile_free]	1.2	Free pages in the swap files
net.if.in[MY_INTERFACE,bytes]	1.0	Incoming traffic on interface MY_INTERFACE
net.if.out[MY_INTERFACE,bytes]	1.0	Outgoing traffic on interface MY_INTERFACE (uppercase parameters denote variable items and must be replaced by actual values or omitted)
proc.num[,,run]	1.0	Number of running processes (com+cur)
proc.num[,,max]	1.0	Max number of processes (idem kernel.maxproc)
proc.num[,,int]	1.0	Number of interactive processes
proc.num[,,batch]	1.0	Number of batch processes
proc.num[,,net]	1.0	Number of network processes
proc.num[,,outswapped]	1.2	Number of out swapped processes (como + hibo + lefo + suspo)
proc.num[,,MODE]	1.0	with MODE in (com, como, cur, hib, hibo, cef, lef, lefo, mwait, pfw, susp, suspo) any other value or no value for MODE will return total number of processes
proc_info[PROCESS_NAME,exists,NODENAME,USERNAME,IMAGNAME]	1.0	Does the process PROCESS_NAME (case blind) on the server ? Returns number of process corresponding to the request. PROCESS_NAME can contain wildcards. NODENAME : the node in the cluster where we try to find the process. Can be * for cluster-wide lookup. Local node only if omitted. USERNAME : username that is running the process. Any username if omitted
proc_info[PROCESS_NAME,imagname,NODENAME,USERNAME,IMAGNAME]	1.2	
proc_info[PROCESS_NAME,nodename,NODENAME,USERNAME,IMAGNAME]	1.2	

proc_info[PROCESS_NAME,username,NODENAME,USERNAME,IMAGNAME]	1.2	
proc_info[PROCESS_NAME,pid,NODENAME,USERNAME,IAMGNAME]	1.2	
proc_info[PROCESS_NAME,prcnam,NODENAME,USERNAME,IMAGNAME]	1.2	
proc_info[PROCESS_NAME,virtpeak,NODENAME,USERNAME,IMAGNAME]	1.2	in KB
proc_info[PROCESS_NAME,wsauth,NODENAME,USERNAME,IMAGNAME]	1.2	in KB
proc_info[PROCESS_NAME,wsauthext,NODENAME,USERNAME,IMAGNAME]	1.2	in KB
proc_info[PROCESS_NAME,wsextent,NODENAME,USERNAME,IMAGNAME]	1.2	in KB
proc_info[PROCESS_NAME,wspeak,NODENAME,USERNAME,IMAGNAME]	1.2	in KB
proc_info[PROCESS_NAME,wsquota,NODENAME,USERNAME,IMAGNAME]	1.2	in KB
proc_info[PROCESS_NAME,wssize,NODENAME,USERNAME,IMAGNAME]	1.2	in KB
proc_info[PROCESS_NAME,biocnt,NODENAME,USERNAME,IMAGNAME]	1.2.2	
proc_info[PROCESS_NAME,biolm,NODENAME,USERNAME,IMAGNAME]	1.2.2	
proc_info[PROCESS_NAME,diocnt,NODENAME,USERNAME,IMAGNAME]	1.2.2	
proc_info[PROCESS_NAME,diolm,NODENAME,USERNAME,IMAGNAME]	1.2.2	
proc_info[PROCESS_NAME,bytcnt,NODENAME,USERNAME,IMAGNAME]	1.2.2	
proc_info[PROCESS_NAME,bytcnt,NODENAME,USERNAME,IMAGNAME]	1.2.2	
proc_info[PROCESS_NAME,enqcnt,NODENAME,USERNAME,IMAGNAME]	1.2.2	
proc_info[PROCESS_NAME,enqlm,NODENAME,USERNAME]	1.2.2	
proc_info[PROCESS_NAME,filcnt,NODENAME,USERNAME]	1.2.2	
proc_info[PROCESS_NAME,fillm,NODENAME,USERNAME]	1.2.2	
proc_info[PROCESS_NAME,prccnt,NODENAME,USERNAME]	1.2.2	
proc_info[PROCESS_NAME,prclm,NODENAME,USERNAME]	1.2.2	
proc_info[PROCESS_NAME,tqcnt,NODENAME,USERNAME]	1.2.2	
proc_info[PROCESS_NAME,tqlm,NODENAME,USERNAME]	1.2.2	
device.errcnt[DEVICE]	1.2	Error count for the DEVICE
vfs.fs.size[DISK,free]	1.0	Free space on disk DISK
vfs.fs.size[DISK,total]	1.0	Total capacity of disk DISK
vfs.fs.shdw[DISK,device_count]	1.1	
vfs.fs.shdw[DISK,mbr_count]	1.1	full member's count (see $GETDVI, DVI$_SHDW_MBR_COUNT)
vfs.file.exists[FILENAME]	1.0	does FILENAME exists (can contain wildcards)
vfs.file.exists[FILENAME,number]	1.0	number of existing FILENAME (can contain wildcards)
vfs.file.size[FILENAME]	1.0	size of FILENAME (or sum of sizes if multiple files due to wildcards)
vfs.file.size[FILENAME,allocated]	1.0	allocated size of FILENAME (or sum...)
queman.queue[QUEUE_NAME,status]	1.2	
queman.queue[QUEUE_NAME,on]	1.2	node_name where QUEUE_NAME is running
queman.queue[QUEUE_NAME,jobs,JOB_STATUS]	1.2	If JOB_STATUS is empty : total number of jobs. If JOB_STATUS in (executing, pending, holding, retained, timed_release) : number of jobs with this status.
queman.queue[QUEUE_NAME,job_limit]	1.2	job_limit for this queue
queman.queue[QUEUE_NAME,description]	1.2	description of this queue
queman.queue[QUEUE_NAME,type]	1.2	type of the queue (BATCH, GENERIC, TERMINAL, PRINTER, SERVER). On shadow DISK (DSAxx), number of shadow members (any state : member, copy, merging). Status of queue QUEUE_NAME (IDLE, PAUSED, PAUSING, RESUMING,STALLED, STARTING, STOPPED, STOPPING, UNAVAILABLE, CLOSED, BUSY,UNDEFINED, AVAILABLE, DISABLED, AUTOSTART_PENDING, STOP_PENDING. Number of jobs in QUEUE_NAME.

queman.job[JOB_NAME,status,QUEUE_NAME]	1.2	Status of job JOB_NAME. The status is in (NO_SUCH_JOB, EXECUTING, STARTING, STALLED, SUSPENDED, ABORTING, PENDING, TIMED_RELEASE, HOLDING, RETAINED). If there are multiple jobs named JOB_NAME, the status is the first status in this list for which there is at least a job. Regarding QUEUE_NAME (same behaviour for all queman.job items). If QUEUE_NAME is empty : look for the jobs in all the queues If QUEUE_NAME in (batch, symbiont, printer, server, terminal), look for jobs in queues of this kind. If something else : it is considered as the name of the queue(s) where to look at for jobs. Can have wildcards.
queman.job[JOB_NAME,count,QUEUE_NAME]	1.2	Total number of jobs named JOB_NAME
queman.job[JOB_NAME,executing,QUEUE_NAME]	1.2	Total number of executing jobs named JOB_NAME
queman.job[JOB_NAME,holding,QUEUE_NAME]	1.2	Total number of holding jobs named JOB_NAME
queman.job[JOB_NAME,pending,QUEUE_NAME]	1.2	Total number of pending jobs maned JOB_NAME
queman.job[JOB_NAME,retained,QUEUE_NAME]	1.2	Total number of retained jobs named JOB_NAME
queman.job[JOB_NAME,stalled,QUEUE_NAME]	1.2	Total number of stalled jobs names JOB_NAME
queman.job[JOB_NAME,starting,QUEUE_NAME]	1.2	Total number of starting jobs named JOB_NAME
queman.job[JOB_NAME,suspended,QUEUE_NAME]	1.2	Total number of suspended jobs named JOB_NAME
queman.job[JOB_NAME,timed_release,QUEUE_NAME]	1.2	Total number of timed released jobs named JOB_NAME
queman.job[JOB_NAME,aborting,QUEUE_NAME]	1.2	Total number of aborting jobs named JOB_NAME
queman.manager[status,QUEUE_MANAGER]	1.2	Status of queue manager (FAILOVER, RUNNING, START_PENDING, STARTING,STOPPING, STOPPED). If QUEUE_MANAGER is empty, it returns a result for the default queue manager.Else, it gets result for the specified queue manager. QUEUE_MANAGER is the name of the queue manager (see : "$ show queue/manager")
queman.manager[node_name,QUEUE_MANAGER]	1.2	node_name where QUEUE_MANAGER is running
queman.manager[nodes,QUEUE_MANAGER]	1.2	list of node names where QUEUE_MANAGER can run status of job JOB_NAME.