

**NAME**

**quota\_open**, **quota\_close**, **quota\_on**, **quota\_off**, **quota\_read**, **quota\_write\_limits**, **quota\_write\_usage**, **quota\_fsname**, **quota\_qfname**, **quota\_maxid**, **quota\_check\_path**, **quota\_convert** - Manipulate quotas

**LIBRARY**

System Utilities Library (libutil, -lutil)

**SYNOPSIS**

```
#include <sys/param.h>
```

```
#include <sys/mount.h>
```

```
#include <ufs/ufs/quota.h>
```

```
#include <fcntl.h>
```

```
#include <fstab.h>
```

```
#include <libutil.h>
```

```
struct quotafile *
```

```
quota_open(struct fstab *fs, int quotatype, int openflags);
```

```
int
```

```
quota_close(struct quotafile *qf);
```

```
int
```

```
quota_on(const struct quotafile *qf);
```

```
int
```

```
quota_off(const struct quotafile *qf);
```

```
int
```

```
quota_read(struct quotafile *qf, struct dqblk *dqb, int id);
```

```
int
```

```
quota_write_limits(struct quotafile *qf, struct dqblk *dqb, int id);
```

```
int
```

```
quota_write_usage(struct quotafile *qf, struct dqblk *dqb, int id);
```

```
const char *
```

```
quota_fsname(const struct quotafile *qf);
```

```
const char *
```

**quota\_qfname**(*const struct quotafile \*qf*);

*int*

**quota\_maxid**(*const struct quotafile \*qf*);

*int*

**quota\_check\_path**(*const struct quotafile \*qf, const char \*path*);

*int*

**quota\_convert**(*struct quotafile \*qf, int wordsize*);

## DESCRIPTION

These functions are designed to simplify access to filesystem quotas. If quotas are active on a filesystem, these functions will access them directly from the kernel using the **quotactl**() system call. If quotas are not active, these functions will access them by reading and writing the quota files directly.

The **quota\_open**() function takes a pointer to an *fstab* entry corresponding to the filesystem on which quotas are to be accessed. The *quotatype* field indicates the type of quotas being sought, either USRQUOTA or GRPQUOTA. The *openflags* are those used by the **open**() system call, usually either O\_RDONLY if the quotas are just to be read, or O\_RDWR if the quotas are to be updated. The O\_CREAT flag should be specified if a new quota file of the requested type should be created if it does not already exist.

The **quota\_close**() function closes any open file descriptors and frees any storage associated with the filesystem and quota type referenced by *qf*.

The **quota\_on**() function enables quotas for the filesystem associated with its *qf* argument which may have been opened with O\_RDONLY or O\_RDWR. The **quota\_on**() function returns 0 if successful; otherwise the value -1 is returned and the global variable *errno* is set to indicate the error, see quotactl(2) for the possible errors.

The **quota\_off**() function disables quotas for the filesystem associated with its *qf* argument which may have been opened with O\_RDONLY or O\_RDWR. The **quota\_off**() function returns 0 if successful; otherwise the value -1 is returned and the global variable *errno* is set to indicate the error, see quotactl(2) for the possible errors.

The **quota\_read**() function reads the quota from the filesystem and quota type referenced by *qf* for the user (or group) specified by *id* into the *dqblk* quota structure pointed to by *dqb*.

The **quota\_write\_limits**() function updates the limit fields (but not the usage fields) for the filesystem

and quota type referenced by *qf* for the user (or group) specified by *id* from the *dqblk* quota structure pointed to by *dqb*.

The **quota\_write\_usage()** function updates the usage fields (but not the limit fields) for the filesystem and quota type referenced by *qf* for the user (or group) specified by *id* from the *dqblk* quota structure pointed to by *dqb*.

The **quota\_fsname()** function returns a pointer to a buffer containing the path to the root of the file system that corresponds to its *qf* argument, as listed in */etc/fstab*. Note that this may be a symbolic link to the actual directory.

The **quota\_qfname()** function returns a pointer to a buffer containing the name of the quota file that corresponds to its *qf* argument. Note that this may be a symbolic link to the actual file.

The **quota\_maxid()** function returns the maximum user (or group) *id* contained in the quota file associated with its *qf* argument.

The **quota\_check\_path()** function checks if the specified path is within the filesystem that corresponds to its *qf* argument. If the *path* argument refers to a symbolic link, **quota\_check\_path()** will follow it.

The **quota\_convert()** function converts the quota file associated with its *qf* argument to the data size specified by its *wordsize* argument. The supported *wordsize* arguments are 32 for the old 32-bit quota file format and 64 for the new 64-bit quota file format. The **quota\_convert()** function may only be called to operate on quota files that are not currently active.

## IMPLEMENTATION NOTES

If the underlying quota file is in or converted to the old 32-bit format, limit and usage values written to the quota file will be clipped to 32 bits.

## RETURN VALUES

If the filesystem has quotas associated with it, **quota\_open()** returns a pointer to a *quotafile* structure used in subsequent quota access calls. If the filesystem has no quotas, or access permission is denied NULL is returned and *errno* is set to indicate the error.

The **quota\_check\_path()** function returns 1 for a positive result and 0 for a negative result. If an error occurs, it returns -1 and sets *errno* to indicate the error.

The **quota\_read()**, **quota\_write\_limits()**, **quota\_write\_usage()**, **quota\_convert()**, and **quota\_close()** functions return zero on success. On error they return -1 and set *errno* to indicate the error.

**SEE ALSO**

quotactl(2), quota.group(5), quota.user(5)

**HISTORY**

The **quotafile** functions first appeared in FreeBSD 8.1.

**AUTHORS**

The **quotafile** functions and this manual page were written by Dag-Erling Smørgrav <des@FreeBSD.org> and Marshall Kirk McKusick <mckusick@mckusick.com>.