NAME

sort_file – Class ss_m Sorting Methods

SYNOPSIS

```
#include <sm_vas.h>  // which includes sm.h

// Logical-id version
static rc_t                 sort_file(
    const lvid_t&               lvid,           // logical vol id (input file)
    const serial_t&             serial,         // serial no for input file
    const lvid_t&               s_lvid,   // logical vol id (output file)
    serial_t&                   s_serial,       // serial no for output file
    store_property_t            property, // property for output file
    const key_info_t&           key_info,       // sort key info
    int                         run_size, // # of pages for each run
    bool                        unique = false, // eliminate duplicates
    bool                        destructive = false); // destroy input file

// Physical-ID version
static rc_t                 sort_file(
    const stid_t&               fid,            // input file
    vid_t                       vid,            // output volume
    stid_t&                     sfid,           // output file (out argument)
    store_property_t            property, // property for output file
    const key_info_t&           key_info, // sort key info
    int                         run_size, // # pages for each run
    bool                        ascending = true,// direction of sort
    bool                        unique = false, // duplicates to be removed?
    bool                        destructive = false, // destroy input file?
    const serial_t&             logical_id=serial_t::null, // assign LID?
                        // for use by logical-ID version

    bool                        use_new_sort = true); // use new sort code?
```

DESCRIPTION

Method **ss_m::sort_file** is an external sorting utility to sort an existing file in the order of specified keys. This API supports the following built-in sort key types: integer, float, char, string and spatial (rectangles on their Hilbert value). Sorting on keys of variable length can only be done if the key is the record header. Sorting on keys of variable location is not supported with this sort implementation.

The user can control the buffer size needed for the sorting by specifying the desired *run_size*. Usually the bigger the *run_size*, the faster the sort.

The caller must provide detailed information on the sort key in *key_info*. This includes information such as type, location and length. The location of the key has to be fixed offset from the beginning of the record (header or body). Class **key_info_t** is further described in **sort_stream_i(ssm).**

Setting the *unique* flag to **true** will eliminate duplicate **records in the file.** (Entire records are compared.) Setting the *destructive* to true will destroy the input file and map the records' logical IDs to those in the new sorted file.

For sorting on variable length records, one must fix the key location by copying the key from body to the beginning of hdr for the record and turn on derived flag in *key_info* before invoking **sort_file.** The result file will have the duplicated key eliminated in each record. Obviously, this works only if the header is unused in the original records (and thus in the result records).

ERRORS

RCOK is returned if there is no lower level error.

EXAMPLES

To sort a file on integer key which is the first field inside each record body, specify the *key_info* as below:

```
key_info.type = t_int;
key_info.where = t_body;
key_info.offset = 0;
key_info.len = sizeof(int);
```

BUGS

No provision is made for byte-swapping keys.

VERSION

This manual page applies to Version 2.0 of the Shore Storage Manager.

SPONSORSHIP

The Shore project is sponsored by the Advanced Research Project Agency, ARPA order number 018 (formerly 8230), monitored by the U.S. Army Research Laboratory under contract DAAB07-91-C-Q518. Further funding for this work was provided by DARPA through Rome Research Laboratory Contract No. F30602-97-2-0247.

COPYRIGHT

SEE ALSO

sort_stream_i(ssm), file(ssm),