NAME

scan_rt_i – Class for Scanning an R*tree index in the Shore Storage Manager

SYNOPSIS

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

class scan_rt_i {
public:

    stid_t              stid;
    tid_t               tid;
    ndx_t               ntype;
    serial_t                  serial;  // serial number if store has
                               // a logical ID

    /* Logical-ID version */
    NORET               scan_rt_i(
      const lvid_t&               lvid,
      const serial_t&       stid,
      sob_cmp_t             c,
      const nbox_t&               box,
      bool                        include_nulls=false,
      concurrency_t               cc = t_cc_page);

    /* Physical-ID version */
    NORET               scan_rt_i(
      const stid_t&               stid,
      sob_cmp_t             c,
      const nbox_t&               box,
      bool                        include_nulls=false,
      concurrency_t               cc = t_cc_page);

    NORET               ~scan_rt_i();

    rc_t                next(
      nbox_t&               key,
      void*                 el,
      smsize_t&             elen,
      bool&                       eof);

    void                finish();

    bool                eof() { return _eof; }
    bool                error_detected()
};
```

DESCRIPTION

Class **scan_rt_i**

TODO

Updates While Scanning

A common question is what is the effect of changes to an index made by a transaction that is also scanning the index. It is not safe to change anything in the file while scanning. Instead, a list of changes should be made during the scan and only performed after the scan is complete.

Null Values

R-trees can contain entries with "null" keys, which are represented by polygons of dimension 0. The data type

```
class nbox_t
```

contains

```
bool nbox_t::is_Null() const;
static nbox_t& nbox_t::Null;
```

for creating and detecting null keys in R-trees.

When scanning an R-tree index, you can skip (default) or collect the entries with "null" keys, according to the value given in the *include_nulls* argument when you create the iterator.

The semantics of a search with nulls is as follows:

inside
: Null is inside everything, including Null.

cover
: Null covers nothing except Null.

overlap
: Null overlaps everything (and everything overlaps null).

ERRORS

To do.

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

Copyright (c) 1994-1999, Computer Sciences Department, University of Wisconsin -- Madison. All Rights Reserved.

SEE ALSO

rtree(ssm), scan_index_i(ssm) intro(ssm),