WEB 3

Type 1 Indexes

IN THIS CHAPTER

- Basic Index Structure

The ability to create indexes on DB2 tables has been around since the first release of DB2. The first type of index that was available is now referred to as a type 1 index. Type 1 indexes were made obsolete when type 2 indexes were introduced in DB2 Version 4. However, IBM supported type 1 indexes through Version 5. As of DB2 V6, type 2 indexes can no longer be used by DB2.

Type 2 indexes are preferable to type 1 indexes because they eliminate index locking. Furthermore, most newer features of DB2 require type 2 indexes.

Basic Index Structure

Before examining the specifics of the layout of index data pages, let's first examine the basic structure of DB2 indexes.

A DB2 index is a modified *b-tree* (balanced tree) structure that orders data values for rapid retrieval. The values being indexed are stored in an inverted tree structure, as shown in Figure 1.

As values are inserted and deleted from the index, the tree structure is automatically balanced, realigning the hierarchy so that the path from top to bottom is uniform. This realignment minimizes the time required to access any given value by keeping the search paths as short as possible. To implement b-tree indexes, DB2 uses the following types of index data pages:

Space map pages	Space map pages determine what space is available in the index for DB2 to utilize.
Root page	Only one root page is available per index. The root page must exist at the highest level of the hierarchy for every index structure. It can be structured as either a leaf or a non-leaf page, depending on the number of entries in the index.

Non-leaf pages	Non-leaf pages are intermediate-level index pages in the b-tree hierarchy. Non-leaf pages need not exist. If they do exist, they contain pointers to other non-leaf pages or leaf pages. They never point to data rows.
Leaf pages	Leaf pages contain pointers to the data rows of a table. Leaf pages must always exist. In a single page index, the root page is a leaf page.

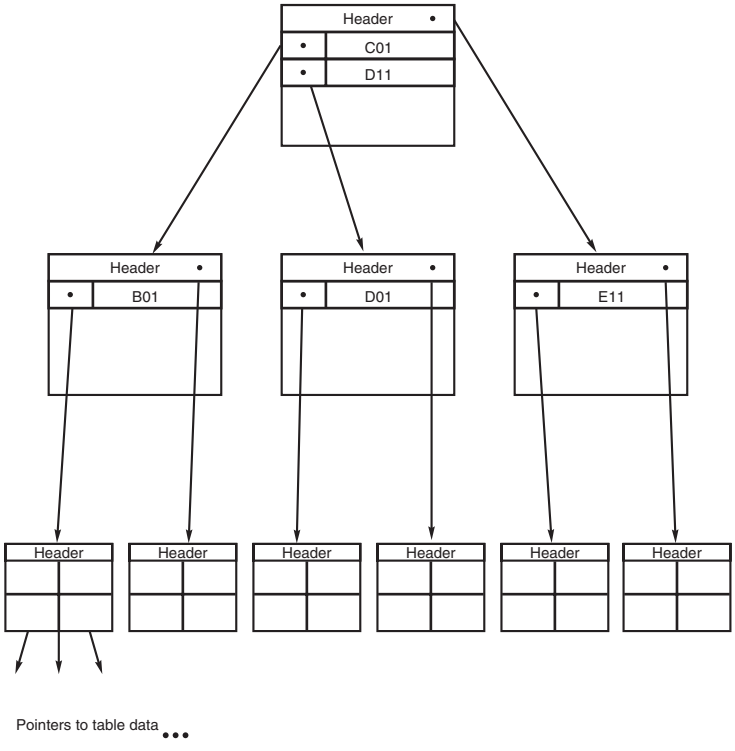

FIGURE 1 DB2 index structure.

The pointers in the leaf pages of an index are called a *record ID*, or *RID*. Each RID is a combination of the tablespace page number and the row pointer for the data value, which together indicate the location of the data value.

The level of a DB2 index indicates whether it contains non-leaf pages. The smallest DB2 index is a one-level index; the root page contains the pointers to the data rows. In this case, the root page is also a leaf page, and no non-leaf pages are available. This is true for Type 1 indexes only; no one-level Type 2 indexes exist. A two-level index does not contain non-leaf pages. The root page points directly to leaf pages, which in turn point to the rows containing the indexed data values.

A three-level index, such as the one shown in Figure 1, contains one level for the root page, another level for non-leaf pages, and a final level for leaf pages. The larger the number of levels for an index, the less efficient it will be. You can have any number of

intermediate non-leaf page levels. Try not to have indexes with more than three levels because they are generally very inefficient.

Type 1 Index Data Pages

Type 1 non-leaf pages are physically formatted as shown in Figure 2. Each non-leaf page contains the following:

- A 12-byte index page header that houses consistency and recoverability information for the index.
- A 16-byte physical header that stores control information for the index page. For example, the physical header controls administrative housekeeping such as the type of page (leaf or non-leaf), the location of the page in the index structure, and the ordering and size of the indexed values.
- A 17-byte logical header that stores additional consistency and recoverability checking information, as well as administers free space.

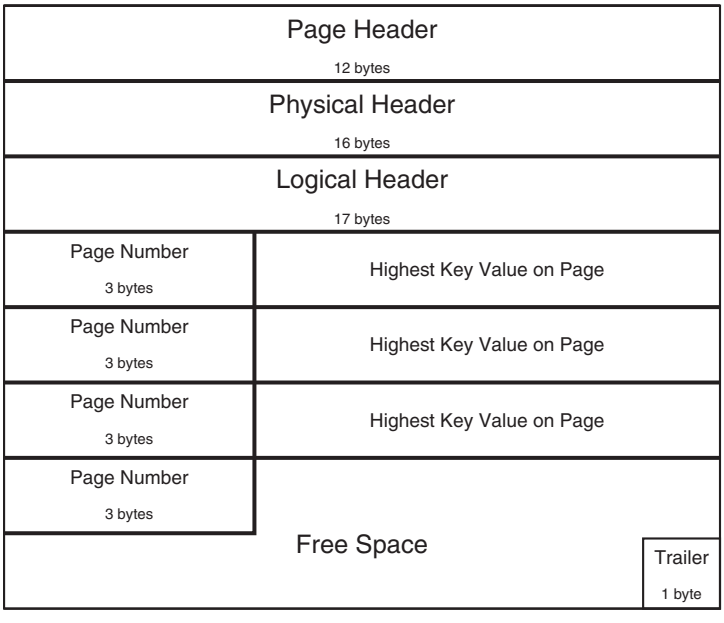

FIGURE 2 Type 1 index non-leaf page layout.

The physical structure of a type 1 index leaf page differs depending on the parameters specified when the index is created. Type 1 index pages can be broken down into smaller portions, known as *subpages*. A type 1 index can be defined as having 1, 2, 4, 8, or 16 subpages. The physical structure of type 1 index leaf pages depends on the number of subpages defined for the index.

For type 1 indexes, increasing the number of subpages can decrease contention, but this may decrease the efficiency of access to the index data. Specify `SUBPAGES 1` for infrequently updated type 1 indexed columns.

For a type 1 clustering index, you might want to try setting the number of subpages such that each subpage contains the same number of rows as the data pages of the tablespace. This can reduce locking of unrelated data. If the index is not clustered, do not attempt this, because the corresponding index subpages will contain different rows than the tablespace pages, and no gain in performance will be realized.

Refer to Figure 3 for the physical layout of a type 1 index leaf page with a subpage specification of 1. The page header, physical header, and logical header are used for the same purposes as they are in non-leaf pages. The remainder of the page is used for index entries. Each index entry is composed of indexed values and RID pointers to the table data.

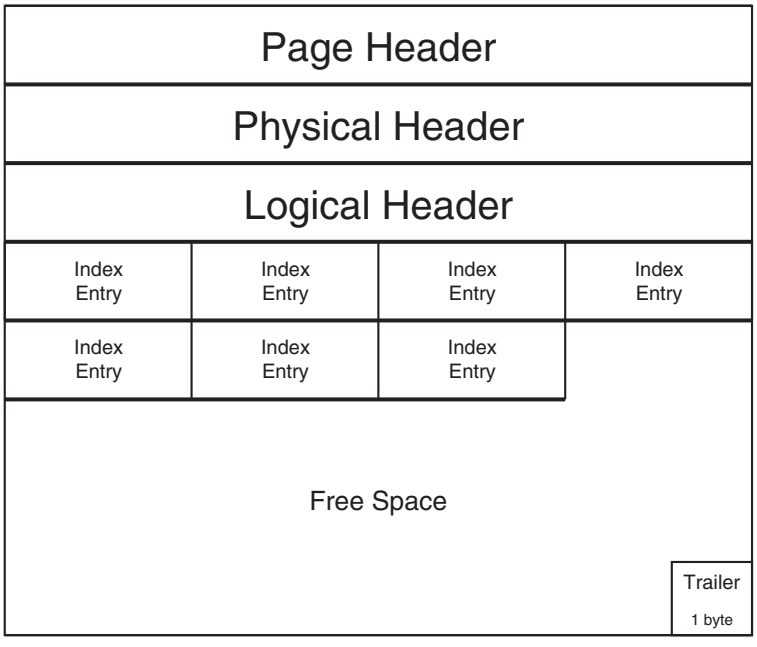

FIGURE 3 Layout of a type 1 index leaf page containing one subpage.

Refer to Figure 4 for the physical layout of a type 1 index leaf page with a subpage specification greater than 1. A subpage directory replaces the single logical header. This directory contains an array of pointers used to locate and administer the index subpages. Each subpage has its own logical header, allowing free space to exist on each subpage.

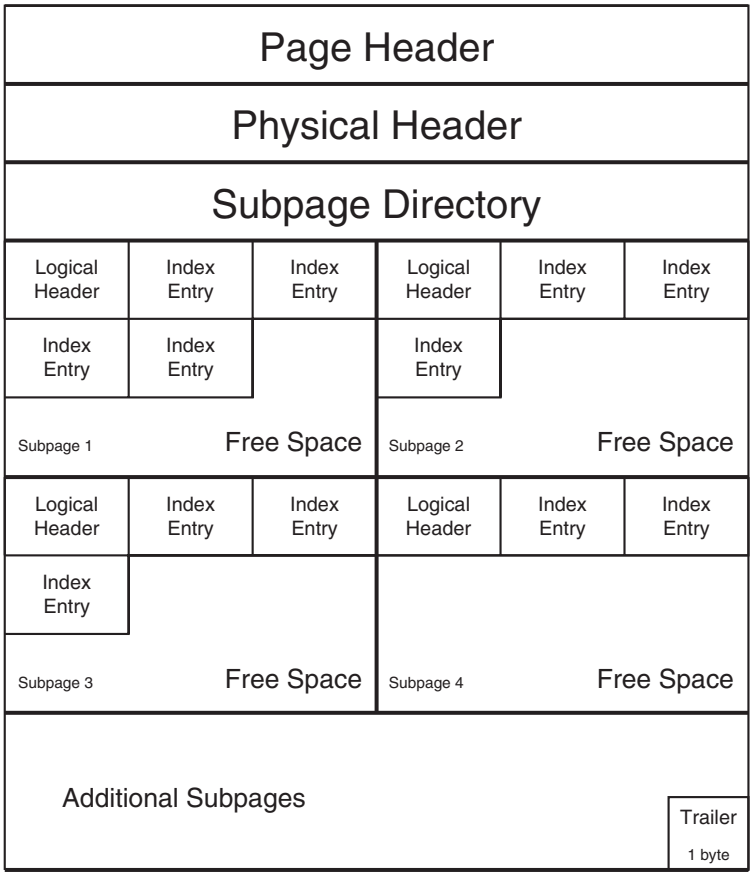

FIGURE 4 Layout of a type 1 index leaf page containing more than one subpage.

The final physical index structure to explore is the index entry. You can create both unique and non-unique indexes for each DB2 table. When the index key is of varying length, DB2 pads the columns to their maximum length, making the index keys a fixed length. A unique index contains entries, and each entry has a single RID. In a unique index, no two index entries can have the same value because the values being indexed are unique (see Figure 5).

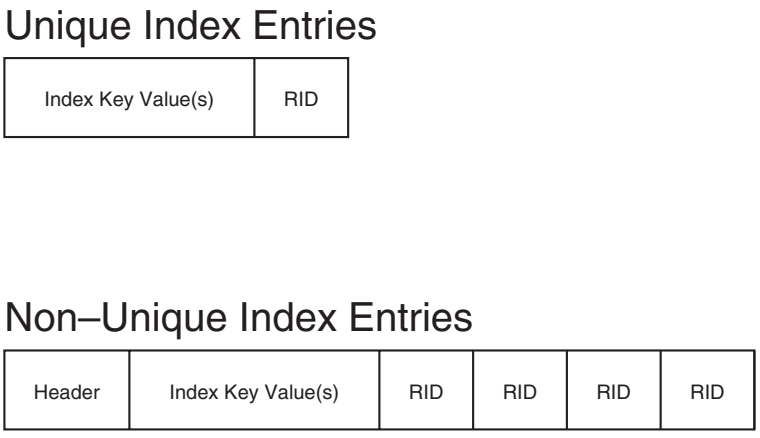

FIGURE 5 Index entries.

Synopsis

This appendix is provided for those shops that have not yet converted to DB2 V6 and still have type 1 indexes. No new indexes should be defined as type 1 and you should immediately begin to convert all type 1 indexes to type 2 indexes. This is important because type 1 indexes are no longer supported by DB2 as of Version 6.