DATA COMPRESSION AND ITS TYPES

6.1. Data Compression

- Compression can reduce the size of a file, improving performance.
- File maintenance can produce fragmentation inside of the file. There are ways to reuse this space.
- There are better ways than sequential search to find a particular record in a file.
- Keysorting is a way to sort medium size files.
- We have already considered how important it is for the file system designer to consider how a file is to be accessed when deciding how to create fields, records, and other file structures. In this chapter, we continue to focus on file organization, but the motivation is different. We look at ways to organize or reorganize files in order to improve performance.
- In the first section, we look at how to organize files to make them smaller. Compression techniques make file smaller by encoding them to remove redundant or unnecessary information.

data compression

The encoding of data in such a way as to reduce its size.

redundancy reduction

Any form of compression which removes only redundant information.

- In this section, we look at ways to make files smaller, using data compression. As with many programming techniques, there are advantages and disadvantages to data compression. In general, the compression must be reversed before the information is used. For this tradeoff,
  - Smaller files use less storage space.
  - The transfer time of disk access is reduced.
  - The transmission time to transfer files over a network is reduced.

  But,
  - Program complexity and size are increased.
  - Computation time is increased.
  - Data portability may be reduced.
  - With some compression methods, information is unrecoverably lost.
  - Direct access may become prohibitably expensive.
  - Data compression is possible because most data contains redundant (repeated) or unnecessary information.

- Data compression is possible because most data contains redundant (repeated) or unnecessary information. Reversible compression removes only redundant information, making it possible to restore the data to its original form. Irreversible compression goes further, removing information which is not actually necessary, making it impossible to recover the original form of the data.
- Next we look at ways to reclaim unused space in files to improve performance. Compaction is a batch process that we can use to purge holes of unused space from a file that has undergone many deletions and updates. Then we investigate dynamic ways to maintain performance by reclaiming space made available by deletions and updates of records during the life of the file.

6.1.1 Compact Notation

The replacement of field values with an ordinal number which index an enumeration of possible field values.

- Compact notation can be used for fields which have an effectively fixed range of values.
- Compact notation can be used for fields which have an effectively fixed range of values. The *State* field of the *Person*record, as used earler, is an example of such a field. There are 676 (26 x 26) possible two letter abbreviations, but there are only 50 states. By assigning an ordinal number to each state, and storing the code as a one byte binary number, the field size is reduced by 50 percent.
- No information has been lost in the process. The compression can be completely reversed, replacing the numeric code with the two letter abbreviation when the file is read. Compact notation is an example of redundancy reduction.
- On the other hand, programs which access the compressed data will need additional code to compress and expand the data. An array can used as a translation table to convert between the numeric codes and the letter abbreviations. The translation table can be coded within the program, using literal constants, or stored in a file which is read into the array by the program.
- Since a file using compact notation contains binary data, it cannot be viewed with a text editor, or typed to the screen. The use of delimited records is prohibitively expensive, since the delimiter will occur in the compacted field.

6.2.2 Run Length Encoding

run-length encoding

An encoding scheme which replaces runs of a single symbol with the symbol and a repetition factor.

- Run-length encoding is useful only when the text contains long runs of a single value.
- Run-length encoding is useful for images which contain solid color areas.

- Run-length encoding may be useful for text which contains strings of blanks.
- Example:
- uncompressed text (hexadecimal format):
- 40 40 40 40 40 40 43 43 41 41 41 41 41 42
- compressed text (hexadecimal format):
- FE 06 40 43 43 FE 05 41 42

  where FE is the compression escape code, followed by a length byte, and the byte to be repeated.

Source : http://elearningatria.files.wordpress.com/2013/10/ise-vi-file-structures-10is63-notes.pdf