G. DIACONESCU
Department of Economic Cybernetics
Academy of Economic Studies

Principles of Organizing and Creating a Statistical Data Base

The paper presents in brief an original procedure of creating inverse files on characters, intensively used in the statistical data base organization. The major point of this method is the BIT (Boolean Inverse Tridimensional) representation of a COBOL type file. Some implementation details are given necessary to understand the total inversion algorithm obtained by the author. The algorithm itself as well as the sorting algorithms, the specifications of a defining language and of a demand language will be published later.

In a number of previous studies [2], [3], [4], the importance of a specific approach to a data base design for statistical data processing has been underlined. In essence, this approach whose basic techniques were defined for the first time in [2] is based on the intense use of inverse files — in a new variant the inversion on characters or the SA-V inversion.

Being based on retrieval algorithms efficiently using the inversion on characters, a statistical data base can take over various functions, such as:

— rapid retrieval at the level of any information element, coming to the retrieval at a character level;
— rapid counting according to various characteristics;
— rapid sorting;
— rapid calculus of histograms;
— rapid retrieval according to value intervals;
— description of data
— creation and flexible maintenance of files;
— representation of some of data tree structures.

The field of application of the statistical data bases is large, taking into account the multitude of the information and management problems implying statistical processing. Processings of a census type (population, assets, inquiries, polls, etc.), statistical technical data (meteo statistics, reliability statistics, etc.), archives data (statistic series in agriculture, demography, etc.), current records of personell, equipment , a.s.o., can be quoted as examples of specific applications.

1. The principles of information retrieval in a statistical data base were shown in [2], but the effective procedure of creating the inverse files on characters was presented in a general way. In the following, implementation details in connection with this problem are given.

The main ideas at the basis of the method designed by the author of the construction of the inverse files on characters are:

1 — the intensive utilization of the properties of purely random access to the internal storage of the electronic computer;

2 — the intensive utilization of the properties of random access on the magnetic disc type physical devices;

3 — the approximation by software of an associative storage in the statistical data base.

The construction of an inverse file on characters implies the existence of an object file organized as a COBOL type sequential file, access being directly through the Record Identity Number (RIN), supposing a COBOL structure identical for all records. The generalization which can be made is to consider an indefinite value for an element of data (a case, frequently found in practice). At the same time, with certain difficulties of implementation, the structure of the COBOL type recording can also be generalized , making possible structure variations from a recording to another.

The object file once created, an inversion can be achieved for each field (attribute) in the structure of the object record, so that eventually, the inverted file of an object represents in fact a collection of attribute inverted files. Inversion according to an attribute allows to solve the retrieval problems according to the respective attribute (key).

The necessary elements for understanding the procedure of creating an inverted file on characters start from the basic notion of objectiv subfile which is illustrated in Fig. 1.

An object subfile may have a common COBOL representation, for example, in EBCDIC character for each field in each recording.

Considering an object subfile, it can be read record by record from the external storage into the internal storage gradually completing an array BIT according to scheme in Fig. 2 where we noted with SA the order number axis of each subattribute (in the sense [2]); we noted with V the axis of possible values of a subattribute (0—9 for numerical fields, for example) and we noted with RIN the axis object record identity number.

The internal storage available for an array BIT dictates the size of the segment and of the logical sector.

The storage space filled in by an array BIT, related to the storage space occupied by an OBJECT subfile represented in characters, is in principle by 50% bigger. This results from calculus:

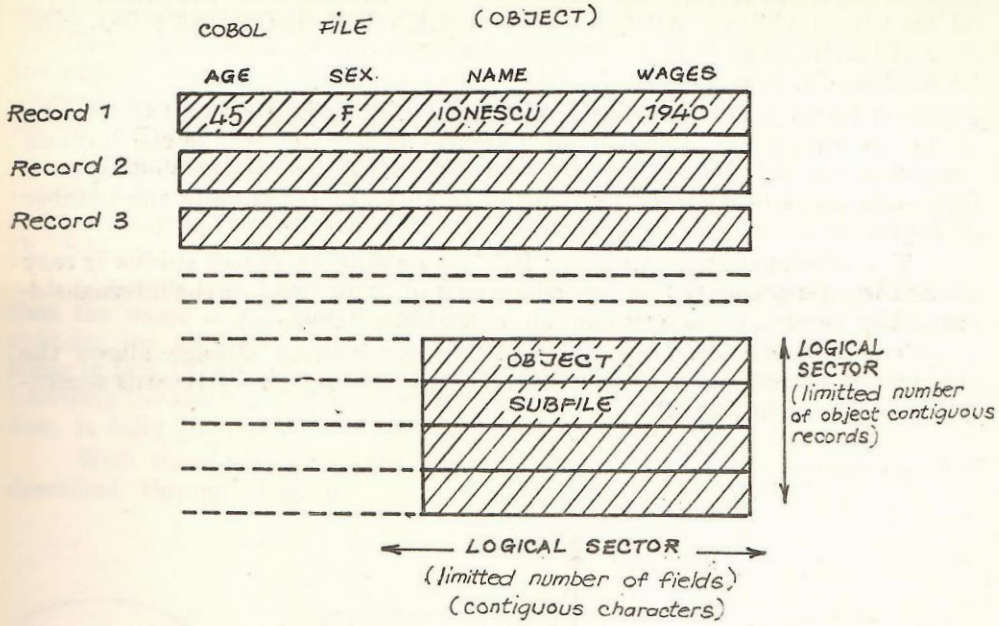
COBOL FILE (OBJECT)
AGE
SEX
NAME
WAGES
Record 1
45
F
IONESCU
1940
Record 2
Record 3
OBJECT
SUBFILE
LOGICAL SECTOR
(limitted number of object contiguous records)
LOGICAL SECTOR
(limitted number of fields)
(contiguous characters)

Fig. 1

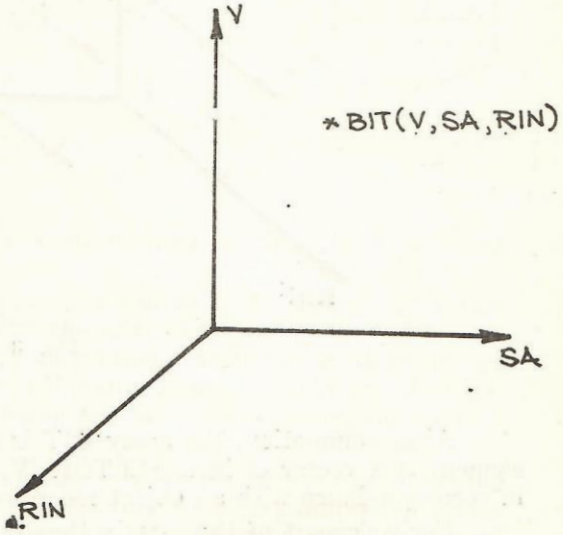
V
* BIT(V,SA,RIN)
SA
RIN

Fig. 2

NUMBER BYTES OBJECT SUBFILE = LENGTH — SEGMENT
LENGTH-SECTOR (IN BYTES) (NUMBER OF RECORDS)
NUMBER BYTES ARRAY BIT = (LENGTH-SEGMENT NO. OF SUBATTRIBUTES) *
LENGTH-SECTOR * 6)* 2/8.

Figure 6 having the significance of V max (taking into account the coding of figures and of the alphabet on 2 figures hexa), figure 2 in the formula signifying the doubling of the segment length because of the coding on 2 figures hexa) and division by 8 being carried out for finding the number of bytes.

The construction of an array BIT by reading an object subfile is easy though the programs require an addressing at a bit level in the internal storage (the easiest to achieve in an assembling language).

The achievement of an array BIT in the internal storage allows the subsequent extraction of all sectors of corresponding inverse records according to scheme in Fig. 3.

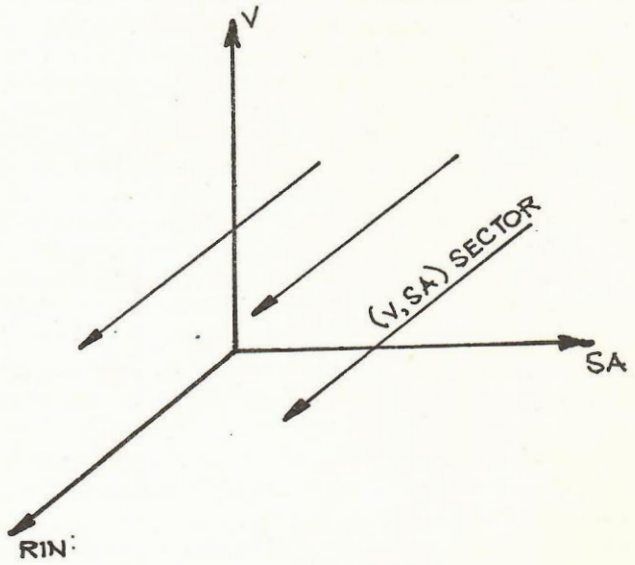

Fig. 3

After completion, the array BIT is considered a matrix in which each element is a vector of bits, SECTOR (V, SA), each bit in the vector being in correspondance with an object recording in the logical sector.

Each element of the matrix thus defines a part of the SA-V inverted record, as it was defined in [2] and [3].

The storing of SECTOR (V, SA) on the magnetic disc containing the inverted file is made by updating a disc preformated with inverted records which contained zero [vector (000000)].

The organization of the preformated inverted file is similar to that of the object, sequential, but just as in FORTRAN a (RIN) order number is attributed to each inverted record, allowing direct access to any inverted record, being given RINV.

Considering a SECTOR (V, SA), the linear address of V and SA is constructed according to the usual formula of linear representation of a matrix (with V varying the fastest), which allows to store on the disc each SECTOR (V, SA).

It results from the way in which the loading of the inverse was described that the name of *a logical track* for a SA-V inverse recording, of a logical cylinder for all logical tracks referring to the same subattribute (always being in fact 12) and of an *inverse subfile* for the part of the inverse completed following the storing of the array BIT from the internal storage on the inverse disc, is fully justified.

With these notions a step from the object inversion procedure can be described through Fig. 4.

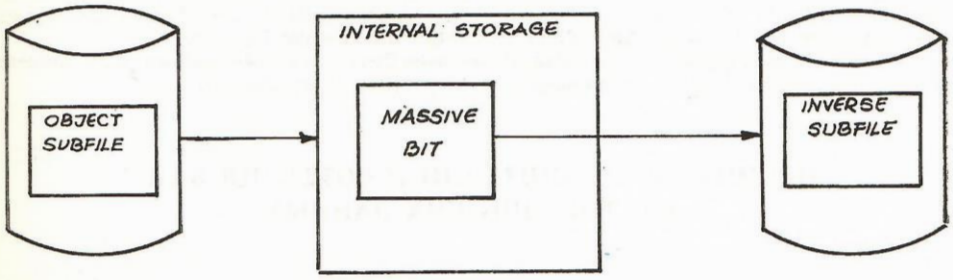

Fig. 4

This step is repeated by a sufficient number of times for exhausting the covering of the object.

The performances of the procedure, measured by a number of access to the magnetic discs, are very good if the relation between the logical segment and the logical sector is conveniently chosen and this relationship can be conveniently chosen in the calculus, because it is only linked to the loading procedure, the only constraint being the quantity of available internal storage.

Practically, a file of approximately 40,000 recordings, each having 1000 bytes can be completely inverted according to all attributes, in about 8 and a half hours on an IBM (370 equipment with an available internal storage of 128 K(for array BIT) and with disc units of 28 million bytes).

REFERENCES

1. IONESCU C. (coordinator) and others: *Dicţionar statistic economic* (Economic Statistic Dictionary), D.C.S., Bucharest, 1969.
2. DIACONESCU G.:*Algoritmul SA-V* (The SA-V Algorithm). Studii şi cercetări de calcul economic şi cibernetică economică, no. 1, 1975.
3. DIACONESCU G.: *Asupra regăsirii informaţiei într-o bază de date statistice* (On Information Retrieval in a Statistic Data Base). Economic Computation and Economic Cybernetic Studies and Research, no 1, 1976.
4. DIACONESCU G.: *Regăsirea informaţiei într-o bază de date organizată SA-V* (Information Retrieval in a SA-V Organized Data Base) The debates of the 3rd International Congress of Cybernetics and Systems (under print).
5. CRICK M.F.C., and others: *A Software Associative Memory for Complex Data Structure.* IBM Corporation, 1970.
6. BLEIFER ROBERT E., and others: *File Organization in TDMS-IFIP.* Congress Edinburgh, 1968.

PRINCIPES D'ORGANISATION ET ETABLISSEMENT *D'UNE BASE DE DONNEES STATISTIQUES*

L'auteur présente de façon succinte une procédure originale pous constituer des fichiers inverses par caractères, employés intensivement à l'organisation des bases de données statistiques. La représentation *BIT* (Boolean Invers Tridimensional) d'un fichier du type Cobol constitue la pointe de cette procedure. Y sont donnés certains détails d'appliation nécessaires à la compréhension de l'algorithme d'inversion totale obtenue par l'auteur.

L'algorithme en propre et les algorithmes assortisseurs, les spécifications d'un langage de définition et d'un langage de demandes seront publiées ultérieurement.

ПРИНЦИПЫ ОРГАНИЗАЦИИ И СОЗДАНИЯ БАЗЫ СТАТИСТИЧЕСКИХ ДАННЫХ

Работа знакомит вкратце с оригинальным приёмом создания обратных знаковых картотек, усиленно используемых при организации баз статистических данных. Центральной точкой этого приёма является представление BIT/
Картотеки типа КОБОЛ. Даются некоторые додробности необходимые для понимания полного обращения алгорифма, осуществлённого автором.

16.02.1976.