Marie Duží; Pavel Materna; Zdenko Staníček
Information capability of a database schema

Kybernetika, Vol. 24 (1988), No. 3, 216--226

Persistent URL: http://dml.cz/dmlcz/125354

KYBERNETIKA — VOLUME 24 (1988), NUMBER 3

INFORMATION CAPABILITY OF A DATABASE SCHEMA

MARIE DUŽÍ, PAVEL MATERNA, ZDENKO STANÍČEK

The power set of the attribute set $\{A_1, ..., A_m\}$, where $A_1, ..., A_m$ constitute "the kernel" of the conceptual schema of a database, can be partially ordered so that

(i) the power set with this partial ordering creates a lattice,

(ii) two other lattices can be derived if the kernel is enriched with attribute sets "informationally equivalent" with subsets of the kernel, and an analogous partial ordering is applied to the resulting set of equivalence classes of attribute sets,

(iii) the partial ordering in all the three above kinds of lattice can be parallelled with the partial ordering of "semantic information".

1. PHILOSOPHY OF HIT DATA MODEL

Data in a database can be viewed as structures; which kind of a structure they are considered to be depends on the data model chosen. Our approach to data structures is a functional one and the model in question is known as HIT Data Model "Homogeneous Integrated Type-oriented"), see [1], [11], [12], [2], [10]. The basic construct of HIT DM is an *attribute*. "Salary of ...", "Children of a given father and a given mother", "Grades of a given student in a given year" are examples of attribute names. Philosophically, attributes are intensions, i.e., they behave as functions the domain of which are states-of-the-world, i.e., pairs (possible world, time point), and which associate every such pair with a function [3], [5]. Formally, they can be presented via attribute names ("attribute identifiers", [1]) the interpretation of which changes dependently on the state-of-the-world.

It is the notion of function (mapping) which makes it possible to define in a most exact way the notion of attribute; therefore, typed lambda-calculus (more precisely, a modification containing "tuple types" – see [1], [7], [12]) has been chosen as the logical apparatus underlying HIT DM. The role of elementary types is played by "*sorts*"; there are two kinds of sorts – *entity sorts* and *descriptive* or representable *sorts*. Omitting the philosophy of sorts in HIT DM [4] one can parallel entity

sorts and descriptive sorts with Chen's "entity sets" and "value sets", respectively [8]. Cartesian products of various sorts are "*tuple types*". Where $S_1, \ldots, S_m$ are sorts, $(S_1, \ldots, S_m)$ is the tuple type composed as cartesian product $S_1 \times \ldots \times S_m$. *Functional types* are sets of functions over the respective types. Thus where T, U are arbitrary types (i.e. tupple types or even functional types), $(T \to U)$ is a functional type, the members of which are (partial and total) functions from T into U.

Attributes are intensions which being applied to a possible world and time point return functions. In HIT DM the attributes which are "data-bearers" are represented by terms of the so-called "*simple types*". Simple types are, properly or philosophically speaking, not the types of attributes but only of those functions which are the result of applying an attribute to a possible world and a time point (w-t-pair). There are two classes of simple types:

a) $(S_1, \ldots, S_m) \to (T_1, \ldots, T_n)$

b) $(S_1, \ldots, S_m) \to ((T_1, \ldots, T_n) \to \text{BOOL})$, $m, n \geqq 1$

where S_i, T_i are sorts and BOOL is the descriptive sort consisting of True, False. The assumption that BOOL is always at our disposal makes it possible to represent such objects as sets (classes) and relations (-in-extension), since classes and relations can be identified with their characteristic functions.

Terms are defined as lambda-terms; two new kinds of a term are added: a *tuple term* $(t_1, \ldots, t_n)$ is a term of a type $(T_1, \ldots, T_n)$, where T_i is the type of the term t_i; a *projection*, $t_{(i)}$, is the ith term of such a tuple term.

A *database schema DS* is defined as the pair (S, CC): S is a tuple of attribute identifiers, the latter are terms $A_1, \ldots, A_n$ of simple types $T_{A1}, \ldots, T_{A_n}$ so that S can be presented as the tuple term $(A_1, \ldots, A_n)$ of the tuple type $(T_{A_1}, \ldots, T_{A_n})$. The particular attribute identifiers are then the particular projections of S, i.e., $A_i = S_{(i)}$.

An interpretation $\mathscr{I}$ of the tuple S is called a *database state*; CC, the so-called "*consistency constraint*", is a term of the type $(T_{A_1}, \ldots, T_{A_n}) \to \text{BOOL}$ which determines the admissible database states, i.e. such interpretations of S in which CC takes the value True.

The logical apparatus for handling various transformations of S is the above characterized modification of the typed lambda calculus using β-rule as the basic rule together with exploiting "logical constants" such as connectives, quantifiers, identities (of the type $(T, T) \to$ BOOL for every type T) and singularizers (of the type $(T \to \text{BOOL}) \to T$ for every type T).

2. INFORMATIONAL EQUIVALENCE

Referring as for exact definitions of the HIT DM concepts to [1], we here reproduce three definitions important for the purpose of the present paper (see [5], [9]).

Definition 1. Let $A, A_1, \ldots, A_m$ be attribute identifiers. We say that the attribute A

is *definable over* the attributes $A_1, \ldots, A_m$ ($A \lhd_{D_1} \{A_1, \ldots, A_m\}$) iff there is a term t such that

(i) t contains occurrences just of $A_1, \ldots, A_m$ and at most some occurrences of variables and logical and/or mathematical constants,

(ii) no subterm of t is an analytic (i.e., logically true or logically false) formula,

(iii) $A = t$ is true in every admissible database state.

Definition 2. The attribute set $\{A_1, \ldots, A_m\}$ is *definable from* attributes $B_1, \ldots, B_n$ ($\{A_1, \ldots, A_m\} \lhd_D \{B_1, \ldots, B_n\}$) iff for any A_i, $1 \leqq i \leqq m$, there is such a subset $\mathscr{B}$ of $\{B_1, \ldots, B_n\}$ that $A_i \lhd_{D_1} \mathscr{B}$.

The relation $\lhd_D$ is reflexive and transitive. Finally:

Definition 3. The attribute set $\{A_1, \ldots, A_m\}$ is *informationally equivalent with* the attribute set $\{B_1, \ldots, B_n\}$ ($\{A_1, \ldots, A_m \approx_i \{B_1, \ldots, B_n\}$) iff $\{A_1, \ldots, A_m\} \lhd_D \lhd_D \{B_1, \ldots, B_n\}$ and $\{B_1, \ldots, B_n\} \lhd_D \{A_1, \ldots, A_m\}$.

The relation $\approx_i$ is an equivalence relation.

Remark. Note that if A is definable from $\{B_1, \ldots, B_m\}$ then $\{A, B_1, \ldots, B_m\} \approx_i \approx_i \{B_1, \ldots, B_m\}$, $m \geqq 1$.

Example. Let A be the attribute named by "The number of children of a given employee", and B the attribute named by "The children of a given employee". A is definable over B: let the respective entity sorts be E (employee), P (person) and the descriptive sort "natural number" be N. Let x, y, z be variables of the type E, P, N, respectively. Then in every admissible database state it holds

$$A = \lambda x\, \imath z(z = \text{Card}\, \lambda y((Bx)\, y))\,,$$

where Card (of the type $(P \to \text{BOOL}) \to N$) is the mathematical function "cardinality of" and "$\imath z$" (an abbreviation of a singularizer) reads "the only z such as". Then, of course,

$$\{B\} \approx_i \{A, B\}\,.$$

3. PARTIAL ORDERING OF (CLASSES OF) ATTRIBUTE SETS

Let $\{A_1, \ldots, A_n\}$ be the set $\mathscr{A}$ of attribute identifiers such that the component S of a database schema DS equals $(A_1, \ldots, A_n)$. It can be shown [5], [9] that the HIT method of designing conceptual schema makes it always possible to choose a (proper or improper) subset $\mathscr{A}'$ of $\mathscr{A}$ such that

(i) no member of $\mathscr{A}'$ is definable over another members of $\mathscr{A}'$,

(ii) every member of $\mathscr{A}'$ is as simple as possible, i.e., "undecomposable" [5], [9],

(iii) $\mathscr{A}' \approx_i \mathscr{A}$.

The subset $\mathscr{A}'$ of $\mathscr{A}$ satisfying (i)–(iii) is called a "*kernel*" of S.

Consider now the kernel $\mathscr{A}' = \{A_1, \ldots, A_n\}$ of some S. Let $P(\mathscr{A}')$ be the power

set of $\mathscr{A}'$. Owing to the fact that $\mathscr{A}'$ satisfies the point (i) of the definition of kernel one can immediately see that the relation $\lhd_D$ is antisymmetric on $P(\mathscr{A}')$. Thus $\lhd_D$ is a partial ordering on $P(\mathscr{A}')$. Moreover, this partial ordering is the same as that induced by the inclusion relation $\subset$. Obviously, $P(\mathscr{A}')$ is a lattice $\mathscr{L}$ with respect to $\lhd_D$; meet and join are the set-theoretical intersection ($\cap$) and union ($\cup$), respectively.

In general, the relation $\lhd_D$ is not antisymmetric – see the above example $\{B\} \approx_i \approx_i \{A, B\}$. A reflexive, antisymmetric and transitive relation defined on the base of $\lhd_D$ can be introduced as follows:

Let $\mathscr{A}$ be any set of attributes. By $!\mathscr{A}!$ we denote the equivalence class generated over $\mathscr{A}$ by the relation $\approx_i$ (i.e., the class of all sets of attributes informationally equivalent with $\mathscr{A}$). We define:

$!\mathscr{B}! \leqq_i !\mathscr{C}!$ iff any member of $!\mathscr{B}!$ is definable from any member of $!\mathscr{C}!$.

The relation $\leqq_i$ – as a relation linking equivalence classes of attribute sets – is antisymmetric.

Thus having a lattice $\mathscr{L}$ with respect to $\lhd_D$ (over $P(\mathscr{A})$, $\mathscr{A}$ a kernel) one can extend $\mathscr{L}$ as follows:

Let $N1, \ldots, N2^n$ be members of $\mathscr{L}$ (i.e., members of $P(\mathscr{A})$, Card $\mathscr{A} = n$). Substituting $!Ni!$ for Ni $(1 \leqq i \leqq 2^n)$ we get a set of equivalence classes constituting a lattice $\mathscr{L}1 = (\{!Nj!\}, \leqq_i)$.

The relation $\leqq_i$ behaves no more as the inclusion relation; meet ($\wedge$) and join ($\vee$) are defined as follows:

$$!\mathscr{A}! \wedge !\mathscr{B}! = !\mathscr{A} \cap \mathscr{B}!\,,$$

$$!\mathscr{A}! \vee !\mathscr{B}! = !\mathscr{A} \cup \mathscr{B}!\,.$$

Another lattice, say $\mathscr{L}_2$, can be obtained as follows:

Let again $\mathscr{A}$ be a kernel-like attribute set. We complement $\mathscr{A}$ by all attributes that are definable over some subset of $\mathscr{A}$ (there can be infinitely many of such attributes, of course). Let the resulting attribute set (which is no more kernel-like) be called $\mathscr{A}_D$. Next consider the power set $P(\mathscr{A}_D)$ of $\mathscr{A}_D$. Take the set of equivalence classes (with respect to $\approx_i$) of those attribute sets which are members of $P(\mathscr{A}_D)$. This set can be partially ordered by $\leqq_i$ analogically as in the preceding case. Again we obtain a lattice (see [5]). Meet and join are defined as follows:

$$!\mathscr{C}! \wedge_i !\mathscr{B}! = !\bigcup_{ij} (\mathscr{C}_i \cap \mathscr{B}_j)!\,,$$

$$!\mathscr{C}! \vee^i !\mathscr{B}! = !\mathscr{C} \cup \mathscr{B}!\,,$$

where $\mathscr{C}_i$ is an arbitrary member of $!\mathscr{C}!$ and $\mathscr{B}_j$ an arbitrary member of $!\mathscr{B}!$.

The former lattice $\mathscr{L}_1$ is a sublattice of this $\mathscr{L}_2$ if they are built up over the same set of attributes.

4. INFORMATION CAPABILITY

Partial ordering $\lhd_D$, linking kernel-like attribute sets, and partial ordering $\leqq_i$, linking equivalence classes of attribute sets, are essentially interconnected. What they have in common is that they realize a partial hierarchy of attribute sets dependent on their "information amount" or "information capability". Indeed, we can use the following formulation:

(IC) $\mathscr{A} \lhd_D \mathscr{B}$ iff the information capability of $\mathscr{A}$ is at most as great as the information capability of $\mathscr{B}$.

$\mathscr{A} \approx_i \mathscr{B}$ iff the information capability of $\mathscr{A}$ is the same as the information capability of $\mathscr{B}$.

Since $!\mathscr{A}! \leqq_i !\mathscr{B}!$ iff $\mathscr{A} \lhd_D \mathscr{B}$, the formulation (IC) can be reformulated using the equivalence classes of attribute sets. Thus the information capability of attribute sets can be compared by means of the respective lattices $\mathscr{L}$, $\mathscr{L}1$, $\mathscr{L}2$.

The formulation (IC) can be motivated by the following consideration.

Attributes are bearers of information. To elucidate this claim imagine an arbitrary (admissible) database state D. An attribute generates in D a class of propositions. Indeed, let A be an attribute identifier of a type $(S_1, \ldots, S_m) \to (T_1, \ldots, T_n)$. Let the term

$$(t_A) \qquad \lambda x_1 \ldots x_m \iota (y_1, \ldots, y_n)\, C\,,$$

where x_i are variables of the type S_i, y_i are variables of the type T_i, and C is a term of the type BOOL, be the term representing A of the given DS. In D, t_A is interpreted due to the data which make D from DS. Applying t_A to particular m-tuples of members ("D-population") of sorts $S_1, \ldots, S_m$ we get terms

$$(t'_A) \qquad \iota(y_1, \ldots, y_n)\, C_j\,, \quad 1 \leqq j \leqq k\,,$$

where C_j is the result of substituting the jth m-tuple from the available m-tuples of S_i-members for $x_1, \ldots, x_m$, the number of such m-tuples in D being k; (t'_A) are terms of the type $(T_1, \ldots, T_n)$. Any (t'_A) "selects" from the n-tuples of members of sorts $T_1, \ldots, T_n$ ("the population" of these sorts in D) that n-tuple which satisfies the respective C_j.

These "double applications" of an attribute identifier can be viewed as giving an information in the respective database state. Thus in any database state an attribute generates propositions, i.e. $\omega \to (\tau \to \mathrm{BOOL})$ – objects (ω – possible worlds, τ – time points). Where w, t are variables of the type "possible worlds", "time points", respectively, and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$, $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n$ are members of the population of sorts $S_1, \ldots, S_m$, $T_1, \ldots, T_n$, respectively, in D, one of the propositions generated in D by A (the "basic proposition") can be constructed as follows:

$$\lambda w \lambda t(((\mathrm{A}w)\, t)\, (\boldsymbol{x}_1. \ldots, \boldsymbol{x}_m)) = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n))$$

(A stands for the respective attribute, i.e., for an intension, here.)

As an example we can adduce the "classical" attribute "Salary of a given employee"

of the type $\omega \to (\tau \to (\text{EMPLOYEE} \to \text{NUMBER}))$ (where $m = 1, n = 1$). Let the data in D be such that the employee Mr. Smith earns \$ 1000 monthly. Let *Sal* be the respective *attribute identifier*. The "double application" of *Sal* can be represented as the term

(Sal_D) $\qquad\qquad$ $(Sal$ 'Mr. Smith) = '1000'.

One of the propositions generated by *the attribute* Sal in D is therefore constructed as follows:

$$\lambda w \lambda t(((\text{Sal } w)\ t)\ \text{Mr. Smith}) = 1000)$$

The proposition constructed in this way is true in those states-of-affairs where Mr. Smith earns montly \$ 1000. Since this is by no means logically necessary, the proposition – and therefore also its encoded representation (Sal_D) – gives a non-trivial information.

The set of "basic propositions" generated in a database state by an attribute A is thus in a way the measure of information which can be obtained from A (in the given database state). (No essential difference arises if A is of a type $(S_1, \ldots, S_m) \to \to ((T_1, \ldots, T_n) \to \text{BOOL})$.) Having a set of attributes, $\{A_1, \ldots, A_n\}$, the "information measure" can be defined (for a database state D) as

$$\bigcup_{i=1}^{n} BP_D^{A_i}$$

where $BP_D^{A_i}$ is the set of "basic propositions" generated by A_i in D.

Let P be an arbitrary set of propositions. By $\mathbf{Cn}P$ we denote the set of all propositions which are implied by P. Then the set of propositions generated by $\{A_1, \ldots, A_n\}$ in D is defined as follows:

$$P_D^{\{A_1, \ldots, A_n\}} = \mathbf{Cn}\left(\bigcup_{i=1}^{n} BP_D^{A_i}\right).$$

Having now the attribute set $\mathscr{A}$, we can immediately see that for any two members $\mathscr{A}_1$, $\mathscr{A}_2$ of $!\mathscr{A}!$ for any database state D

$$P_D^{\mathscr{A}_1} = P_D^{\mathscr{A}_2}.$$

It is also clear that if $!\mathscr{A}! \leqq_i !\mathscr{B}!$ and $!\mathscr{A}! \neq !\mathscr{B}!$ then

$$P_D^{\mathscr{A}} \subset P_D^{\mathscr{B}} \quad \text{and} \quad P_D^{\mathscr{A}} \neq P_D^{\mathscr{B}}$$

for any database state D.

It was Carnap, who together with Bar-Hillel has proposed the idea of measuring semantic information in terms of possible worlds being eliminated by a given proposition [6]. Taking over this very intuitive idea we can say that a set of propositions which eliminates more possible worlds "bears" more semantic information than a set of propositions which eliminates less possible worlds. Thus a proper subset P' of a set of propositions P bears "less" semantic information than P, unless, of course, $P \setminus P' \subset \mathbf{Cn}(P')$. It should be, however, immediately clear that if $!\mathscr{A}! \leqq_i !\mathscr{B}!$ and $!\mathscr{A}! \neq !\mathscr{B}!$ then

$$P_D^{\mathscr{B}} \setminus P_D^{\mathscr{A}} \not\subset \mathbf{Cn}(P_D^{\mathscr{A}})$$

for at least one D. (Indeed,

$$\mathbf{Cn}(P_{\mathcal{G}}^{\mathcal{A}}) = \mathbf{Cn}\left(\mathbf{Cn}\left(\bigcup_{i=1}^{n} BP_D^{A_i}\right)\right) = \mathbf{Cn}\left(\bigcup_{i=1}^{n} BP_D^{A_i}\right) = P_D^{\mathcal{A}}$$

and $P_D^{\mathcal{A}} \neq P_D^{\mathcal{B}}$.)

Thus the partial ordering of (equivalence classes of) attribute sets can be parallelled with the partial ordering of sets of propositions (generated in a database state by the respective attribute sets) induced by the relation of logical implication, i.e., consequence relation.

5. WHAT IS THIS THEORY GOOD FOR?

In Section 1 a database schema has been defined as the pair (S, CC), where S is a tuple of attribute identifiers, CC the so-called consistency constraint. In this section we will outline the employment of the theory of information capability in the design of a database system following the principles of the three-level architecture [13]. All the schemas (conceptual, internal and external) are supposed to satisfy

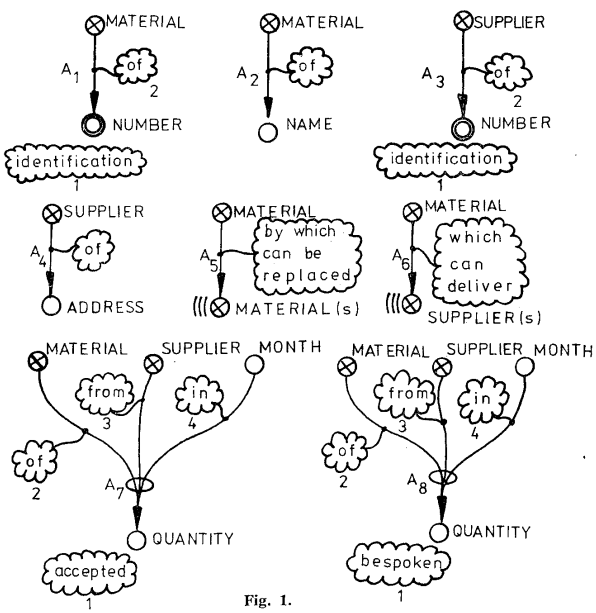

Fig. 1.

the above definition of a schema (cf. [11], [12]). Stating a natural presumption that the information capability of a schema is determined by the attributes which are described in the schema we have explored information capability of attribute sets. For the sake of briefness the set of attributes, identifiers of which form the first component S of a conceptual (external, internal) schema, will be in the following text referred to as the conceptual (external, internal) set of attributes.

Example. Consider the set $S_1 = \{A_1, A_2, \ldots, A_8\}$ of attributes of Fig. 1. This set can be considered to be the conceptual set of attributes.

The information capability of S_1 is determined by the class $!S_1!$. Substituting attributes A'_7, A'_8 (see Fig. 2) for attributes A_7, A_8, respectively, we obtain the set

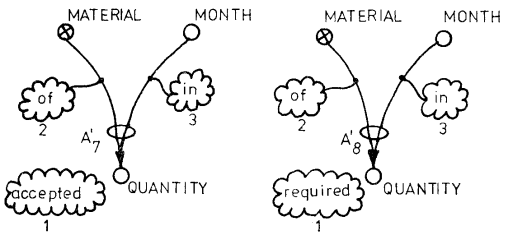

Fig. 2.

$S_2 = \{A_1, \ldots, A_6, A'_7, A'_8\}$ which is informationally weaker than S_1. Indeed, attribute A'_7 is definable over A_7, not vice versa. Denoting *mat*, *mon*, *sup*, *q* variables of types MATERIAL, MONTH, SUPPLIER, QUANTITY, respectively, we can specify A'_7:

$$A'_7 = \lambda\, mat\ mon\ q(q = \sum q'(\exists\, sup\,(A_7\ mat\ sup\ mon) = q'))$$

Similarly A'_8 is definable over A_8.

Another informationally weaker set S_3 of attributes can be obtained from S_1 when replacing A_7, A_8 by A''_7, A''_8:

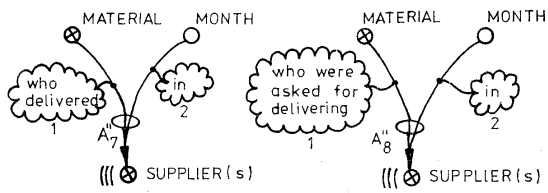

Fig. 3.

The attribute A_7'' is again definable over A_7:

$$A_7'' = \lambda\, mat\ mon\ \lambda\, sup\ \exists q((A_7\ mat\ sup\ mon) = q \wedge q > 0)$$

Analogically for A_8''.

The information capabilities of the sets S_2, S_3 are not comparable, for attributes A_7', A_8' are not definable over any subset of S_3 and vice versa A_7'', A_8'' are not definable over any subset of S_2.

Now consider the set $S_4 = \{A_1, \ldots, A_6, A_7''', A_8'''\}$, where A_7''', A_8''' are attributes from Fig. 4. Obviously, this set S_4 is definable both from S_2 and S_3.

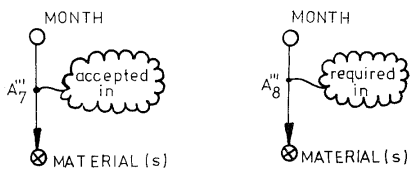

Fig. 4.

Hence

$$S_4 \lhd_D S_2 \lhd_D S_1\,,$$

$$S_4 \lhd_D S_3 \lhd_D S_1$$

and the equivalence classes generated over these sets form the following lattice:

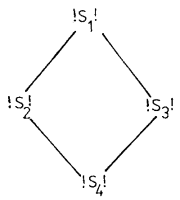

Fig. 5.

In this very simplified example of a database system the lattice illustrating ordering of information capability may seem to be rather artificial. (Besides, we used only a very simple way of "deriving" attributes.) Such an information lattice should be, however, found in every real database application and it may be worth realizing the following facts concerning this lattice:

a) The conceptual set of attributes as well as the internal set of attributes must be members of the "highest" class, i.e., they must possess the greatest information capability (not regarding distributed database systems in which internal schemas "mirror" the particular external schemas rather than the conceptual one).

b) While the conceptual schema – as an invariant of the database system – is the kernel (cf. [5]) of all the members of this class, the internal schema (though informationally equivalent with the conceptual one) may contain some definable attributes, i.e., a controlled degree of redundancy of the stored data may be useful. Though reducing the degree of redundancy decreases demands on disc storage and facilitates updates, some informationally redundant attributes may be found in the internal schema. They are such attributes which are definable over the kernel (conceptual) attributes by means of such mathematical/logical functions which are not effectively realizable with software and hardware at our disposal. The redundancy may also enhance the reliability of the system using the redundant information to reconstruct the database state which has been lost by an error.

c) The particular external sets of attributes are members of "lower" classes. The descending ordering of these sets mirrors the ascending ordering of the particular management levels of an enterprise. The "lowest" class contains the external view of the highest management level on which highly aggregated information is required. The schemas of different classes on the same level are not comparable, for they describe different business applications.

Concluding we may say that a breach of the above principles signals a disorder in the management system and/or its automation. The objective of this paper is therefore to work out a theory which can, e.g., provide tools for controlling efficiency and correctness of an information management system.

(Received September 24, 1987.)

REFERENCES

[1] J. Zlatuška: HIT data model. Databases from the functional point of view. Proc. VLDB '85, Stockholm, 1985, 470–477.

[2] F. Krejčí and Z. Staníček: HIT methodology for complicated data structure design. Proc. 8th Internat. Seminar on DBMS, Piešťany, 1985, 217–235.

[3] P. Materna and J. Pokorný: Applying simple theory of types to data bases. Information Systems *6* (1981), 4, 283–300.

[4] P. Materna: Entity Sorts: What are they? Computers and Artificial Intelligence *6* (1987), 4, 321–324.

[5] M. Duží, Fr. Krejčí, P. Materna and Z. Staníček: HIT Method of the Database Design. Research Report, Techn. University Brno, Prague 1986, pages 77.

[6] R. Carnap and Y. Bar-Hillel: An Outline of Semantic Information. Technical Report No. 247, MIT, Research Laboratory in Electronics, 1952. Reprinted in: Y. Bar-Hillel: Language and Information, Addison-Wesley and the Jerusalem Academia Press, 1964.

[7] J. Zlatuška, T. Chrz and P. Materna: Lambda-calculus (in Czech). Proc. SOFSEM '85, 1985, 195–242.

[8] P. P. S. Chen: The entity-relationship model: Towards a unified view of data. ACM Trans. on Database Systems *1* (1976), 1, 9–36.

[9] M. Duží and Z. Staníček: A functional approach to the representation of information in the HIT method (in Czech). Proc. Datasem '86, Znojmo 1986, part, 2, 20–29.

[10] Z. Staníček and F. Krejčí: The contribution to the IDMS data base design based on the HIT data model. Proc. 7th Internat. Seminar on DBMS, Varna, October 1984, 34–46.

[11] J. Zlatuška: HIT data model. A functional approach to data bases. Proc. 7th Internat. Seminar on DBMS, Varna, October 1984, 21—33.
[12] O. Felix and J. Zlatuška: Transforming external queries into internal operations with data using HIT data model. Proc. 8th Internat. Seminar on DBMS, Piešťany, October, 1985.
[13] D. Tsichritzis and A. Klug (eds.): The ANSI(X3)SPARC DBMS framework. Report of Study Group on Database Management Systems. Information Systems *3* (1975), 1, 173—191.

RNDr. Marie Duží, ČKD Tatra, Plzeňská 6, 150 40 Praha 5. Doc. Dr. Pavel Materna, CSc., PÚDIS (Design Institute), Lidových milicí 69, 112 76 Praha 1. RNDr. Zdenko Staníček, INORGA (Institute for Industrial Management), Na Františku 32, 110 00 Praha 1. Czechoslovakia.