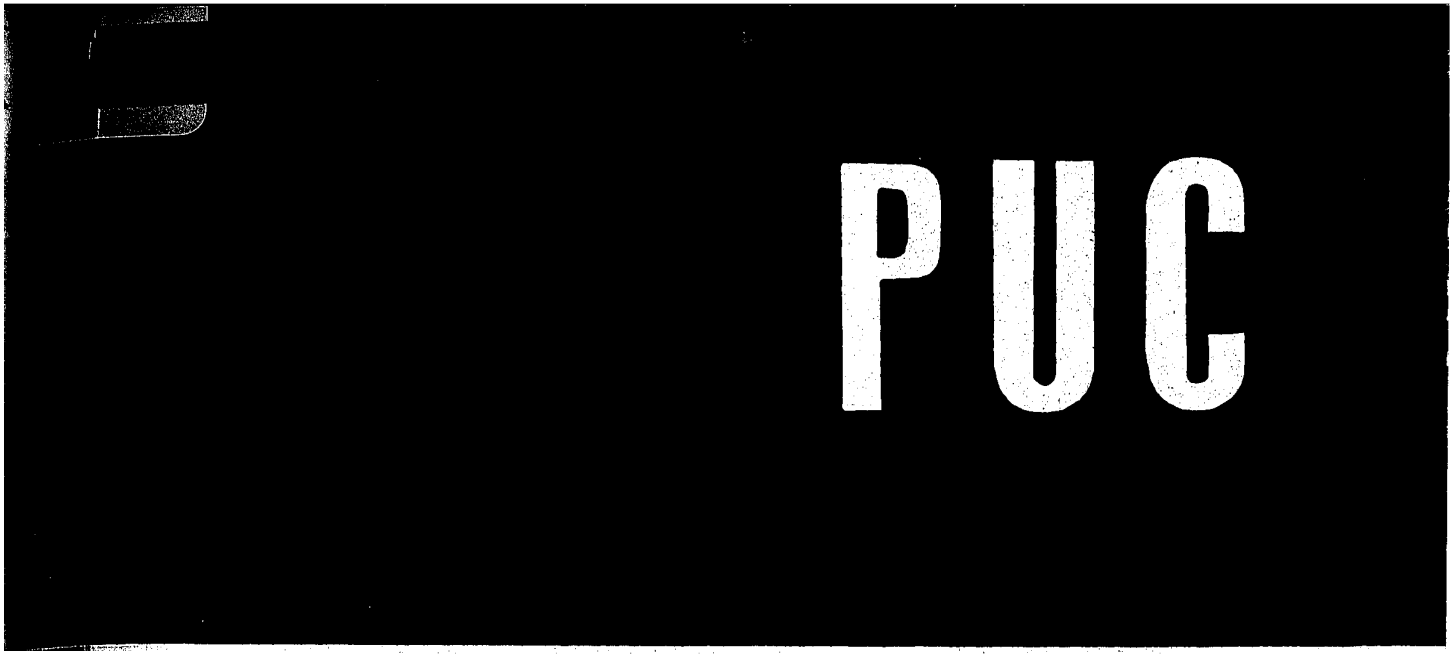


Series: Monographs in Computer Science
and Computer Applications.

Nº 2/76

A FUNCTIONAL DATA BASE MODEL

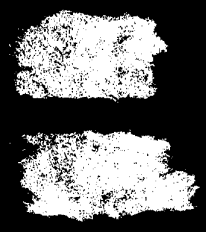
by

Larry Kerschberg

and

João E.S. Pacheco

Departamento de Informática



Pontificia Universidade Catolica do Rio de Janeiro
Rua Marques de São Vicente, 209 — ZC 20
Rio de Janeiro — Brasil

Series: Monographs in Computer Science
and Computer Applications

Nº 2/76

A FUNCTIONAL DATA BASE MODEL*

by

Larry Kerschberg

and

João E.S. Pacheco**

Series Editor: Sergio E.R. Carvalho February, 1976

*This research was sponsored in part by the Brazilian Government Agency FINEP under contract Nº 244/CT, and developed by the Data Base Group.

**Companhia Real de Processamento de Dados
Rua Santo André, 62, São Paulo, SP - Brasil.

Copies may be requested from :

Rosane T.L. Castilho, Head
Setor de Documentação e Informação
Depto. de Informática - PUC/RJ
Rua Marquês de São Vicente, 209 - Gávea
20.000 - Rio de Janeiro - RJ. BRASIL.

ACKNOWLEDGEMENTS

We would like to thank Dr. H. H. Wedekind and R. Nascimento Melo for the helpfull comments. Special thanks go to J. A. Mirabile who made several fundamental contributions to this work.

ABSTRACT:

A Functional model of data is presented as a labelled pseudograph whose nodes are sets and whose arcs are total functions. The model allows one to represent partial functions,binary relations, n-ary relations, as well as m-ary associations among relations. Injective functions play the role of candidate keys.

Graph algorithms transform the model into the relational DBTG/CODASYL, and entity set models. Query schema are defined as queries in normal form and can be specified via quantified semi-paths in the graph. A Query Schema Syntax is proposed for query specifications.

KEY WORDS:

Data base specification, graph transformations , functional model, relational model, query schema.

RESUMO:

Um modelo funcional para dados é apresentado como um pseudografo rotulado, cujos nós são conjuntos e cujas arestas são funções totais. O modelo permite a representação de funções parciais, relações binárias, relações n-arias, assim como associações m-arias entre relações. Funções injetivas desempenham o papel de chaves possiveis.

Algoritmos de grafos transformam o modelo no modelo relacional DBTG/CODASYL, e em modelos de entidades/conjuntos. Esquemas de consultas são definidos como consultas em forma normal e podem ser especificados por meio de caminhos quantificados no grafo. Uma sintaxe para esquemas de consulta é proposta.

PALAVRAS CHAVE:

Especificação de banco de dados, tranformações de grafos, modelo funcional, modelo relacional, esquemas de consulta.

CONTENTS

1 - INTRODUCTION

An examination of extant logical data base models shows that all of them essentially relate data items to other data items. Moreover, these relationships not only determine the nature of the model but also influence the thought processes involved in obtaining it. Both the DBTG/CODASYL [1] and Senko's Entity Set/ DIAM II [2] models relate data items in one-to-one, one-to-many , and many-to-many relationships. In the former, data items are organized into data structures called record types, sets, and networks, respectively. In the Entity Set model, all binary relations and their converses are specified between those data items which serve as unique identifiers for entities and all other data items to which they are related.

Codd's Relational Model [3] organizes data items into a finite collection of named n-ary relations, and the data items are called attributes. Codd introduces the notion of functional dependence [4] among attributes, and shows that relations can be placed in optimal third normal form to avoid undesirable insertion, deletion, and modification dependencies. Wang and Wedekind [5]propose a method of synthesizing logical segments based on: functional dependencies among attributes, system performance requirements , and minimal covers for the collection of relations obtained.

Recently, Schmid and Swenson [6] have proposed a "semantic" graph model which can be viewed as a network of independent object types, each of which is described by its "characteristics". They provide a characterization theorem for relations in third normal form obtained from the model.

One would expect a certain degree of similarity among the models, and this is indeed the case. Nijssen [7] maintains that after some modifications to both approaches, relational and CODASYL data bases and languages could work together. Further, Tsichritzis [8] suggests implementing the relational model via the network approach using "links".

In this study we propose a formal mathematical framework for the specification of data base models. The Functional Data Base Model (FDBM, Functional Model) is basically a graph model whose vertices represent sets and whose arcs are total functions. The sets may be interpreted as n-ary relations, entity sets, or record types. Furthermore, the FDBM designer may create vertices which represent either real - world entities or artificial sets which permit a better understanding of data item interrelationships.

We have found that the functional model has the following advantages:

1. Systems Analysts find it easy to understand and use;
2. Functional dependencies among data items are easily determined;
3. The other models mentioned can be obtained by graph algorithms; and
4. Query schema can be specified quite naturally via semipaths in the graph.

2 - MATHEMATICAL PRELIMINARIES

We begin by presenting relevant concepts from Algebra [9] and Graph Theory [10]. It is assumed that the reader is familiar with the notion of a set as well as the union, intersection, and difference of sets. The reader will note that in defining a binary relation we make a distinction between its name and its set of ordered pairs. Moreover, the composition of relations and functions will have the same notation.

2.1 - SETS, RELATIONS, AND FUNCTIONS

Let A and B be sets. A is a subset of B, $A \subseteq B$, if every element of A is an element of B. Sets A and B are equal, $A = B$, if they have the same elements, and are disjoint, $A \cap B = \phi$, if they have no elements in common. The notation $\{a \in A \mid P\}$ means the set elements of A which satisfy property P.

The cartesion product of sets A and B, A x B, is the set of ordered pairs $A \times B = \{(a,b) \mid a \in A \text{ and } b \in B\}$. A binary relation α from A to B is the subset $R_\alpha \subseteq A \times B$. The converse α^c, of α is $R_\alpha c = \{(b,a) \mid (a,b) \in R_\alpha\}$. The domain of α, $\Delta(\alpha) = \{a \in A \mid \exists b \in B : (a,b) \in R_\alpha\}$. Let α and β be binary relations ($R_\alpha \subseteq A \times B$ and $R_\beta \subseteq B \times C$). The composition, $\alpha \circ \beta$, of α and β is binary relation from A to C such that

$$R_{\alpha \circ \beta} = \{(a,c) \mid \exists b \in B : (a,b) \in R_\alpha \text{ and } (b,c) \in R_\beta\}$$

A binary relation α from A to B is a (total) function, $\alpha : A \to B$, if for each $a \in A$ there is at most one $b \in B$ such that $(a,b) \in R_\alpha$. The relation α is a partial function when $\Delta(\alpha) \subset A$, and α restricted to $\Delta(\alpha)$ is a function.

A function $\alpha : A \to B$ is injective (1:1, an injection) if for $a \neq a'$ in A then $\alpha(a) \neq \alpha(a')$ in B; it is surjective (onto, a surjection) if for each $b \in B$, there is at least one $a \in A$ such that $b = \alpha(a)$; it is bijective (a bijection) if it is 1:1 and onto. If $\alpha : A \to B$ is injective (bijective), then the relation α^{-1} from B to A such that $R = \{(b,a) \mid b = \alpha(a)\}$ is a partial [total] function called the inverse of α.

Let $A_1, A_2, \ldots, A_n$ be sets and let $\underset{n}{X} A$ denote their cartesian product. An n-ary relation α on $\underset{n}{X} A$ is a subset $R_\alpha \subseteq \underset{n}{X} A$. An element of R_a is called an n-tuple or tuple.

We now present a result which will be used extensively in the Functional Model. Let $A^S = \{f \mid f : S \to A\}$ be the set of functions from S to A. There exists a bijection (see [9]) between the function sets $(A \times B)^S$ and $A^S \times B^S$. Thus, for every $h \in (A \times B)^S$ there exists a pair of functions $(f, g) \in A^S \times B^S$ such that $f = h \circ p$ and $g = h \circ q$, where p and q are surjections such that $p: A \times B \to A$ and $q: A \times B \to B$.

Fact 2.1 Let α be a binary relation from A to B. The injection (inclusion) $h : R_\alpha \to A \times B$ admits a decomposition into a pair of functions (f,g) such that $f = h \circ p$, $g = h \circ q$ with p and q as defined above.

We note that the result can be extended to n-ary relations as follows. If α is an n-ary relation such that $R_\alpha \subseteq \underset{n}{X} A$, then the injection $h : R_\alpha \to \underset{n}{X} A$ admits a decomposition into n functions $(f_1, f_2, \ldots, f_n)$ where $f_i = h \circ \pi_i$ and $\pi_i : \underset{n}{X} A \to A_i$ is a surjection.

In the sequel the decomposition $(f_1, f_2, \ldots, f_n)$ of a relation will be called its functional specification. The next result characterizes various types of binary relations in terms of their functional specifications.

Fact 2.2 Let α be a binary relation from A to B with functional specification (f,g). The following statements hold true:

1. If f is 1:1 (g is 1:1), then α (α^c) is a partial function.
2. If f is bijective, then α is a total function.
3. If f is bijective and g is 1:1 (onto, bijective), then α is a function which is 1:1 (onto, bijective).

2.2 - DIGRAPHS, PATHS, AND SEMIPATHS

A digraph $D = (V, A)$ consists of a finite set V of vertices or nodes and a set A of ordered pairs of distinct vertices. A pair (v,w) is called an arc and may be written as vw. The arc goes from v to w, v and w are said to be adjacent, and the arc is said to be incident with v and w. The outdegree, $od(v)$, of a vertex v is the number of nodes adjacent to it.

A digraph is said to be labelled if its vertices and/or arcs are labelled. A digraph is a multigraph if more than one arc is incident with a pair of nodes v and w, and a pseudograph if it is a multigraph with self-loops.

A (directed) walk in a digraph is an alternating sequence of vertices and arcs, v_0, a_1, v_1, a_2, v_2, ..., a_n, v_n, such that $a_i = v_{i-1}\, v_i$. A cycle is a non-trivial closed walk ($v_o = v_n$) with all other vertices distinct. The number of arcs in a path is its length. A geodesic from v to w is a path from v to w of minimum length.

A semiwalk is an alternating sequence of vertices and arcs, v_0, a_1, v_1, a_2, v_2, ..., v_n, in which an arc may be traversed contrary to its orientation. The concepts of semipath and semicycle are similar to those of path and cycle.

Node v is said to be reachable from node w if these exists a path from w to v in D. Node w is said to be joinable to v if there is a semipath from w to v. The set $R(w) = \{v \in V \mid$ v is reachable from w$\}$ is called the reachable set of node w. The set $J(w) = \{v \in V \mid$ w is joinable to v$\}$ is called the joinable set of node w. In general, $R(w) \subseteq J(w)$ for $w \in V$.

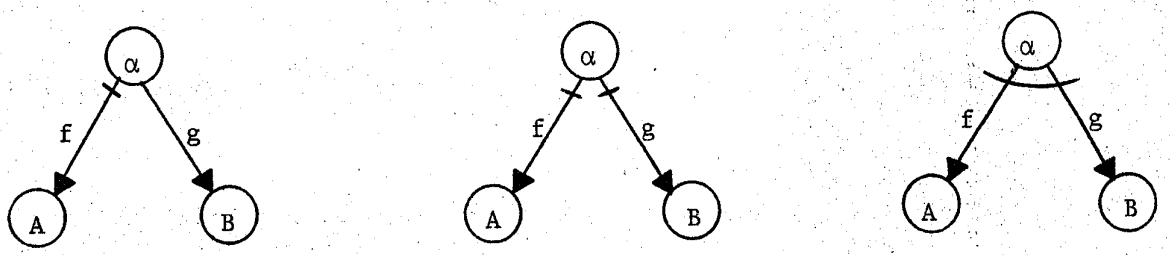
In subsequent sections we present the Functional Model, graph algorithms which transform our model into the relational , CODASYL, and entity set models, a normal form for queries, and an informal query language systax based on the model.

3 - THE FUNCTIONAL DATA BASE MODEL

In this section we define the Functional Model as a pseudograph whose nodes are sets and whose arcs are total functions. We show that this framework is rich enough to model partial functions, binary relations, n-ary relations, and m-ary "associations" among relations. Injective functions play an important role in the model, and are comparable to candidate keys in the relational model.

The starting point for the model is the Fact 2.2 that a binary relation α from A to B admits a functional specification (f,g) such that $f: R_\alpha \to A$ and $g: R_\alpha \to B$.

For the Functional Model three types of functional specifications are important: f is injective, f and g are injective, and both f and g when taken together, form an injection. These cases are shown in Figure 1a and correspond respectively to α being a partial function, α and α^c (its converse) are partial functions, and α is a relation (f and g taken together represent the inclusion $R_\alpha \rightarrow AxB$). The bar across an arc denotes an injective function.



α is a partial function (a)

α and α^c are partial functions (b)

α is a relation (c)

Important Functional Specifications

Figure 1a.

We note that in figure 1a, α is the name of the vertex, while its contents is the set of ordered pairs, R_α. Since all tuples in R_α are distinct, a unique identifier can be associated with each one, thereby allowing us to refer to each element η of α. Thus $f(\eta)$ and $g(\eta)$ are function values which describe η. Cases a,b and c of figure 1a can be interpreted as follows:

Case a) If f is injective then it has an inverse f^{-1} such that $\alpha = f^{-1} \circ g$. Thus if $a \in A$ is a component of a tuple (a,b) of R_α, then it is a component of exactly one such tuple so that b is obtained by applying g.

Case b) If both f and g are injective, then $\alpha = f^{-1} \circ g$ and $\alpha^c = g^{-1} \circ f$. Thus, knowledge of either component of a tuple in R_α is sufficient to determine the other one.

Case c) If α is a relation then knowledge of both components of a tuple is required to access each tuple of R_α. In this case $\alpha = f^c \circ g$ and $\alpha^c = g^c \circ f$. We say that the composite (f,g) is injective as it represents the inclusion $R_\alpha \rightarrow AxB$.

Now we present a formal definition of the Functional Model, together with an example and comments.

Definition 3.1 A Functional Data Base Model is a labelled pseudograph FDBM = (V,A) such that

1. Each vertex of V represents a set of entities of the application being modelled, and has a distinct label called its name. A special vertex C is used to denote the set of character strings generated by the particular alphabet used.

2. Each arc in the non-empty set A represents a total function, and has a label which is its name. The arc labels need not be distinct, but arcs should be distinguishable within the context of the vertices to which they are incident. Those arcs with codomain C are depicted (see Figure 2*) as pointing into space, for clarity of graphical representation.

We assume that a data definition facility for the model would appropriately type functions with codomain C.

Example 3.2 This example is based on a model of a firm developed by P. Fehder [12] for his 99 Queries. We now present each set together with its functional specification. Figure 2 depicts the FDBM for this model.

1. The set EMP (Employees) has functional specification (MNO, NAME, ADDR, JOB, SAL, MGR, WORKS IN) which corresponds respectively to the employee's man number (injective), name, address, job number, salary, manager, and the department in which he works.

2. The set PROJ (Projects) has functional specification (PNO, NAME, RSEC, EMP_COUNT, MGR) which corresponds respectively to the project number, name, security rating, the number of employees working on the project, and the project manager. Moreover PNO is an injective function.

3. The set DEPT (Departments) has functional specification (DNO, NAME, LOC, EMP_COUNT, MGR) which corresponds respectively to the department number (injective), name, location, employee count, and manager.

* See the Appendix

4. The set PART has functional specification (PNO, NAME, NWT, SHWT, VAL) which correspond respectively to the part number (injective), name, net weight, shipping weight and dollar value.

5. The set EP (Employees-in-Projects) has functional specification (PEP, EPE, EMP_STATUS) with the composite (PEP, EPE) being injective. Note that PEP: EP → PROJ, EPE: EP → EMP, and EMP_STATUS is the employee's status with respect to a given project in which he is working.

6. The set MNFG (Manufacturing) has functional specification (DM, MP, M_PERIOD, U_COST, NPARTS_MANF) where DM: MNFG → DEPT, MP: MNFG → PART, and the composites (DM,MP) and(MP,M_PERIOD) are injective. Moreover, the last three functions of the specification are the manufacturing period in quarters, the unit cost of production, and the number of parts manufactured, respectively.

7. The set ASSYOP (Assembly Operations) has functional specification (PRIMARY, COMPONENT, NCOM_PARTS) with injective composite (PRIMARY, COMPONENT). A given primary part may have any number of component parts and may itself be a component of any number of parts, but may not be a component of itself or one of its components.

4 - TRANSFORMATIONS OF THE FUNCTIONAL MODEL

In this section graph theoretical concepts are used to transform a FDBM to its equivalent relational model, DBTG/CODASYL model and Entity Set / DIAM II model.

Algorithm 4.1 The Functional to Relational Transform (Figure 3)

Let F(v) denote those functions in the specification of v with codamain other than C.

1. Obtain a list L of those $v \in V - C$ with indegree zero.

2. Choose a vertex v' from L. For each $g \in F(v')$ find the shortest path from v' to C whose first arc is g and whose last arc is an injective function. This path p = v',g,v'',.....,f, C replaces g, which is deleted once all paths involving g have been found (see Remark 4.2 below).

3. Delete v' from L, and if v" is of indegree zero then add it to L.

4. Repeat steps 2 and 3 until all v ε V - C have been examined.

Remark 4.2 Since a FDBM may model a hierarchy of relations (unnormalized relations in Codd's terminology), it is quite possible for vertex v" in p to have an injective composite (h,k) as part of its functional specification. In this case shortest paths p_1= v", h, ..., C and p_2= v", k, ..., C must be obtained and paths v', g, p_1 and v', g, p_2 appended to v'.

The paths obtained in the algorithm represent the composition of functions, hence they are themselves functions. The name of a path p is the concatenation of the arc names comprising it, but in unambignous cases one such arc name is usually sufficient.

Notice that the resulting relational model is itself a FDBM which is devoid of the semantic content and functional consistency constraints provided by the original FDBM. Further, the relations obtained are in third normal form.

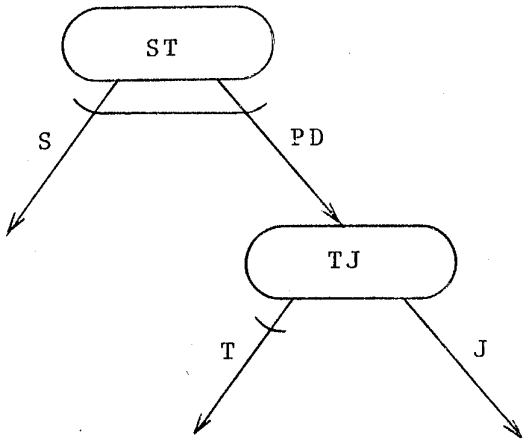
To illustrate this point we present an example given by Date [16]. Consider a model for student-names (S), course-names (J), and teacher-names (T) such that the following semantic rules apply:

1. For each subject, each student of that subject is taught by only one teacher.

2. Each teacher teaches only one subject .

3. Each subject is taught by several teachers.

Figure 1b below summarizes the semantic rules in terms of the Functional Model.



Student, Subject, Teacher Model

Figure 1b

The model was obtained by using rule 1 to create a vertex STJ with injective composite (S, T). Rules 2 and 3 state that subjects (T) are solely dependent on teachers (T), so we create a new vertex TJ with appropriate functions T and J and push it down from STL (renamed ST) via function PD. Thus, ST is a relation while TJ is a function. Application of algorithm 4.2 to the above model will yield two relations in third normal form. They are ST and TJ with attributes, S,T and T,T, respectively.

The Functional Model construction methodology forces the systems analyst to understand the semantics of the application in much the same way that structured programming forces the analyst to really understand the algorithms he is programming.

Algorithm 4.3 The Functional to DBTG/CODASYL Transform (Figure 4)

1. Given a FDBM obtain its dual, $FDBM^d$, by reversing the direction of all arcs.

2. Each vertex $v \in V - C$ is a record type.

3. For a given $v \in V - C$ its records are composed of data items corresponding to arcs from C to v.

4. The CODASYL "sets"* are precisely those arcs incident to vertices in V - C.

5. The keys which access records are the injective funtions of the FDBM. The vertices with injective composites in the FDBM are "networks" in the CODASYL model.

Algorithm 4.4 The Functional to Entity Set/DIAM II Transform (Figure 5).

1. For each $v \in V - C$, consider the function set $A(v) = \{f: v \to C\}$. For every $f \in A(v)$, create a new vertex v' whose name is that of f, such that there exists a path $v \to v' \to C$.

2. Delete vertex C from the FDBM.

3. For each vertex $v \in V$ whose injective function $f: v \to v'$ is not a composite, rename v as the concatenation of names v and v', and then delete v' (and its incident arc) from the FDBM.

4. The Entity Set Model is obtained by taking the union of the FDBM and its dual, $FDBM^d$, ie.,

   Entity Set Model = FDBM U $FDBM^d$ = $(V, A \cup A^d)$.

   where A^d denotes the set of arcs with their directions reversed, and appropriate names assigned to them (see Figure 5).

Remark 4.5 The entity set model associates each entity set with its unique identifier and relates these identifiers to other data items via binary relations and their converses. For example, the binary relation EMP_MNO → NAME of Figure 5 corresponds to the composition of functions, $MNO^{-1} \circ NAME$, in the FDBM of Figure 2, while the converse relation

NAME → EMP_MNO is $(MNO^{-1} \circ NAME)^C = NAME^C \circ MNO$ in the FDBM.

---

* This algorithm assumes "recursive" sets which are presently not allowed in the CODASYL model because of implementation restrictions.

5 - QUERY SPECIFICATION IN THE FUNCTIONAL MODEL

In this section we show how the Functional Model can be used to specify queries. Since the FDBM is a graph model which reflects the semantics of data item interrelationships, one would expect a query to correspond to a walk through the graph.

Informally, query specification in the Functional Model involves the enumeration of a sequence of quantified semipaths. These semipaths form an AND/OR graph which can be converted into an AND/OR "tree" with possible loops by spliting vertices having more than one incoming arc.

In order to specify queries to the FDBM we need a language which takes advantage of the most basic properties of the model. In our case they are that:

1. The graph provides an overview of the semantic and topological data relationships.
2. For querying purposes, each arc of the FDBM can be considered a binary relation, which may be traversed in either direction.
3. Queries are highly context-dependent, and the semipaths built up by walking through the graph reflect this dependence.

In his study of elementary programs, Engeler [13] defines a normal form for programs based on program composition and a looping operation. Since a query can be viewed as a relation (to be discussed shortly), then query composition correspond to relation composition while the looping operation is used for certain queries.

In general, a query to the FDBM is a relation, rel (C,T) from C to T, where T is a singleton set, called the truth value. Thus, a query relates vertices of the FDBM in a complicated fashion, and the user wants to know which values (in C) of certain specified functions satisfy the relation. The relation, rel(C,T) is itself the composition of other relations which correspond to the quantified semipaths mentioned above. The first step in query specification is to refine rel (C,T) to rel(C,X) $\circ$ rel(X,T) where X is a vertex of the FDBM other than C. The relation rel(C,X) denotes those functions of X to be output, while rel(X,T) is a relation which qualifies which

elements of X are to be selected.

With this brief introduction we now informally define the syntax of a query language based on the FDBM. For the sake of brevity only the retrieval aspects are presented. However, the AQUERIUS language [14] has capabilities for modification queries, built-in functions, arithmetic expressions, and has facilities for set, function, and relation definitions.

Now we proceed to define a query schema to the FDBM via a query schema syntax. Notice that bracketed terms are optional, an asteriks denotes a repeated term, and keys denote that any of the several terms may be chosen.

Definition 5.1 A query schema to a FDBM is a query specified by means of the following Query Schema Syntax (QSS). Let X,Y and Z be distinct vertices of a FDBM = (V^*, A) with $V^* = V \cup T$.

query → WHAT ARE THE rel(C,T)?

rel(X,Y) → arc_name(X,Y) [quantifier]
|rel(X,Z) [Z-name] rel(Z,Y)
|NOT rel(X,Y)
|(rel(X,Y)[,rel(X,Y)]* { AND / OR / ALSO } rel(X,Y))
|(rel(X,Y))

quantifier → EACH|ALL|ONLY|ONLY ALL|SOME

Special arc-names:

rel(C,C) → EQUAL|GREATER THAN|LESS THAN
|GREATER THAN OR EQUAL|LESS THAN OR EQUAL

rel(C,T) → character string constant

rel(X,T) → ITSELF|HIMSELF|[WHATEVER]

Notation

1. The keyword ALSO is used when specifying which vertex functions are to be output. The sequence of relations separated by commas will be considered as a conjunction or a disjunction of relations if the keywords AND or OR are invoked, respectively.

2. The quantifiers ALL, ONLY and ONLY ALL correspond, respectively, to A implies B($A \supseteq B$), B implies A($A \subseteq B$) and A if and only if B($A = B$). SOME is the default quantifier used in relation composition.

3. The quantifier EACH means for each (for all, $\forall$) and is used to create hierarchical output tables, as does GROUPED BY in SEQUEL [15].

4. The keywords ITSELF and HIMSELF create loops in a schema as show in QSS1 and QSS2 helow. When these words are used, the codomain set need not be repeated.

5. The keyword WHATEVER may be used to relate a vertex to the truth value, T, although a comma after the vertex name will usually be sufficient.

6. Note that for query specification, each function of the FDBM is regarded as binary relation, since it may be traversed in either direction. When an arc is traversed contrary to its orientation, the converse relation name should be used. In this presentation, however, we use the function name and let the vertex sequencing determine whether the converse relation has been invoked.

7. Upon reaching the vertex C, the user must invoke a special arc--name, rel(C,C), which can then be followed by a quantifier, rel(C,T), or rel(C,Z) where $Z \varepsilon V^*$. Notice that the C-name is not invoked.

The following English and QSS queries are based on the model in Figure 2.

1. What are the names of employees whose managers earn less than they do?

   QSS1. WHAT ARE THE NAME EMP(MGR EMP SAL LESS THAN SAL) EMP HIMSELF?

2. What are the names and numbers of projects whose security ratings (RSEC) are strictly less the job numbers of all employees assigned to them?

   QSS2. WHAT ARE THE (NAME ALSO PNO) PROJ RSEC LESS THAN ALL JOB EMP EPE EP PEP PROJ ITSELF?

3. What are the names and numbers of parts, manufactured by departments which employ more than 100 people and are located in New York, whose component parts all have netweight greater than 10lbs.?

QSS3. WHAT ARE THE (PNAME ALSO PNO) PART (MP MNFG DM DEPT(LOC EQUAL 'NEW YORK' AND EMP_COUNT GREATER THAN 100) AND PRIMARY ASSYOP COMPONENT ALL PART NWT GREATER THAN 10)?

4. What are the names, numbers, and addresses of employees who work in departments which only manufacture parts of unit cost less than $100 and whose component values are all greater than $10?

QSS4. WHAT ARE THE (NAME, MNO ALSO ADDR) EMP WORKS IN DEPT DM ONLY MNFG (U_COST LESS THAN 100 AND MP PART PRIMARY ASSYOP COMPONENT ALL PART VAL GREATER THAN 10)?

Remark 5.2 The four non-trivial queries were chosen to illustrate how the QSS form follows almost directly from the English.

The set, function, and relation defining facilities of AQUERIUS [14] permit a higher degree of naturalness and flexibility in query specification. We note the vertices such as EP, MNFG and ASSYOP represent transitive verbs involving adjacent vertices. Thus, the sequences EPE EP PEP, MP MNFG DM, and PRIMARY ASSYOP COMPONENT correspond to EMP "assigned to" PROJ, PART "manufactured by" DEPT, and PART "with components" PART, DEPT "manufactures" PART, and PART "used in" PART. Moreover, relative pronouns and prepositions such as WHICH, WHO, WHOSE, WITH, IN and OF may be added to the syntax to make the queries more "natural".

The four queries would then look like:

QSS1. WHAT ARE THE NAME [OF] EMP [WHOSE] (MGR EMP SAL LESS THAN SAL) [OF] EMP HIMSELF?

QSS2. WHAT ARE THE (NAME ALSO PNO) [OF] PROJ [WITH] RSEC LESS THAN ALL JOB [OF] EMP "assigned to" PROJ ITSELF?

QSS3. WHAT ARE THE (PNAME ALSO PNO) [OF] PART ("manufactured by" DEPT [WITH] (LOC EQUAL 'NEW YORK' AND EMP_COUNT GREATER THAN 100) AND "with components" ALL PART [OF]NWT GREATER THAN 10)?

QSS4. WHAT ARE THE (NAME, MNO ALSO ADDR) [OF] EMP [WHO] WORKS IN DEPT DM ONLY MNFG (U_COST LESS THAN 100 AND MP PART "with components" ALL PART [OF] VAL GREATER THAN 10)?

6 - CONCLUSIONS

The Functional Data Base Model provides a simple yet formal methodology for the design of logical data models. We believe that systems analysts can easily think of total functions when modelling the semantics of a particular application, since each vertex of the model is completely determined by its outgoing arcs.

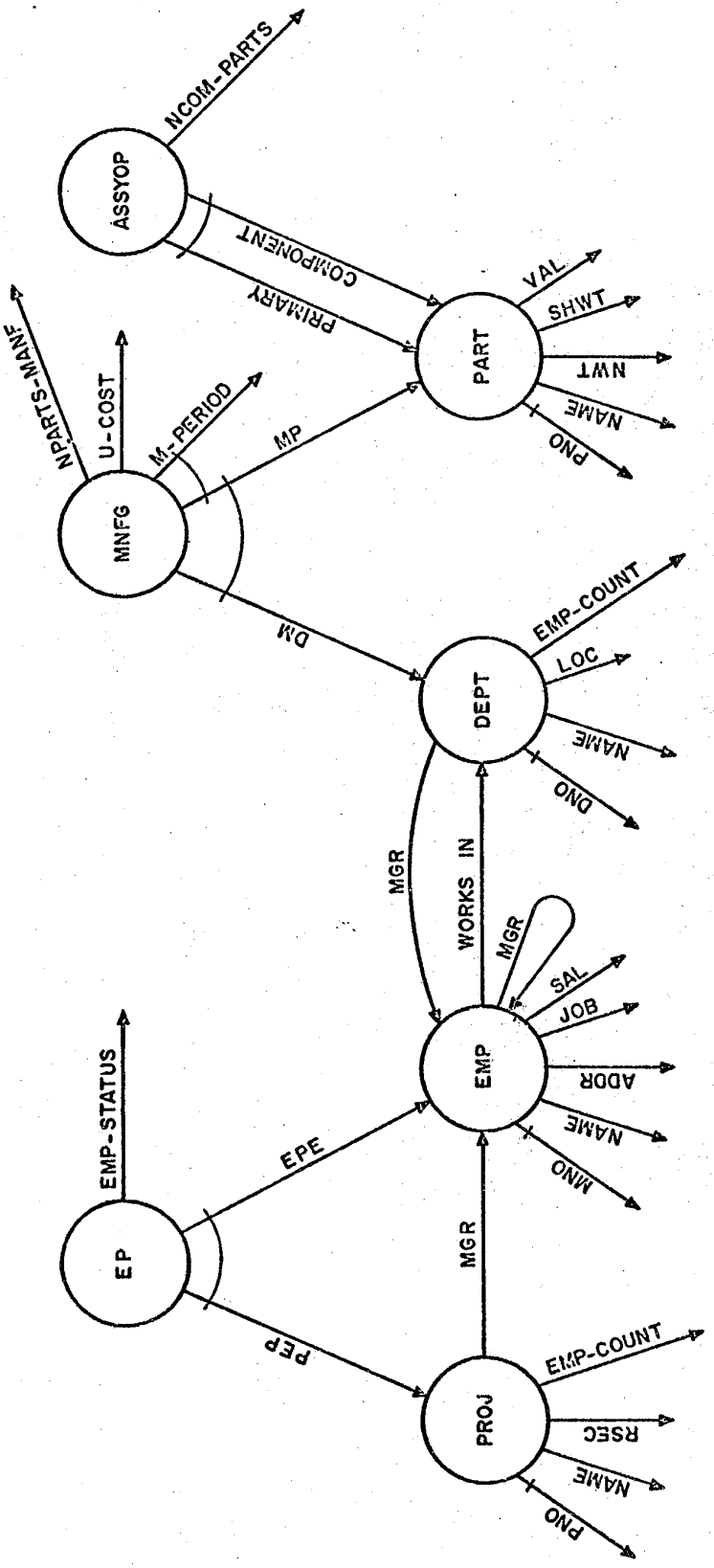
Graph Algorithms can be applied to the Functional Model in order to obtain the relational, DBTG/CODASYL, and entity set models, Further, the relational model is itself a FDBM striped of its "semantics", ie., the functions incident to vertices of V-C. The CODASYL model is the topological dual of the FDBM, while the entity set model can be informally viewed as the union of the modified FDBM and its dual, $FDBM^d$.

The Functional Model provides a convenient framework for query specification. The query schema syntax allows complex queries to be developed by simply walking through the graph. The sample queries presented show how the specification parallels the English version. This language appears more "natural" than say SEQUEL, and for querying purposes the system, not the user, handles set operations such as union, intersection, difference and subsetting which correspond to the boolean operations OR, AND, NOT and the quantifiers ALL, ONLY, and ONLY ALL.

A query schema generates an AND/OR graph which must be processed. In abstract terms this processing involves assigning the truth value, T, to those elements of a set which satisfy the corresponding qualification. Further, the quantifiers ALL, ONLY, and ONLY ALL, when invoked, demand that certain subsetting criteria be satisfied. Lastly, we note that the parallelism of the AND/OR graph could be used to optimize the performance of the query processor.

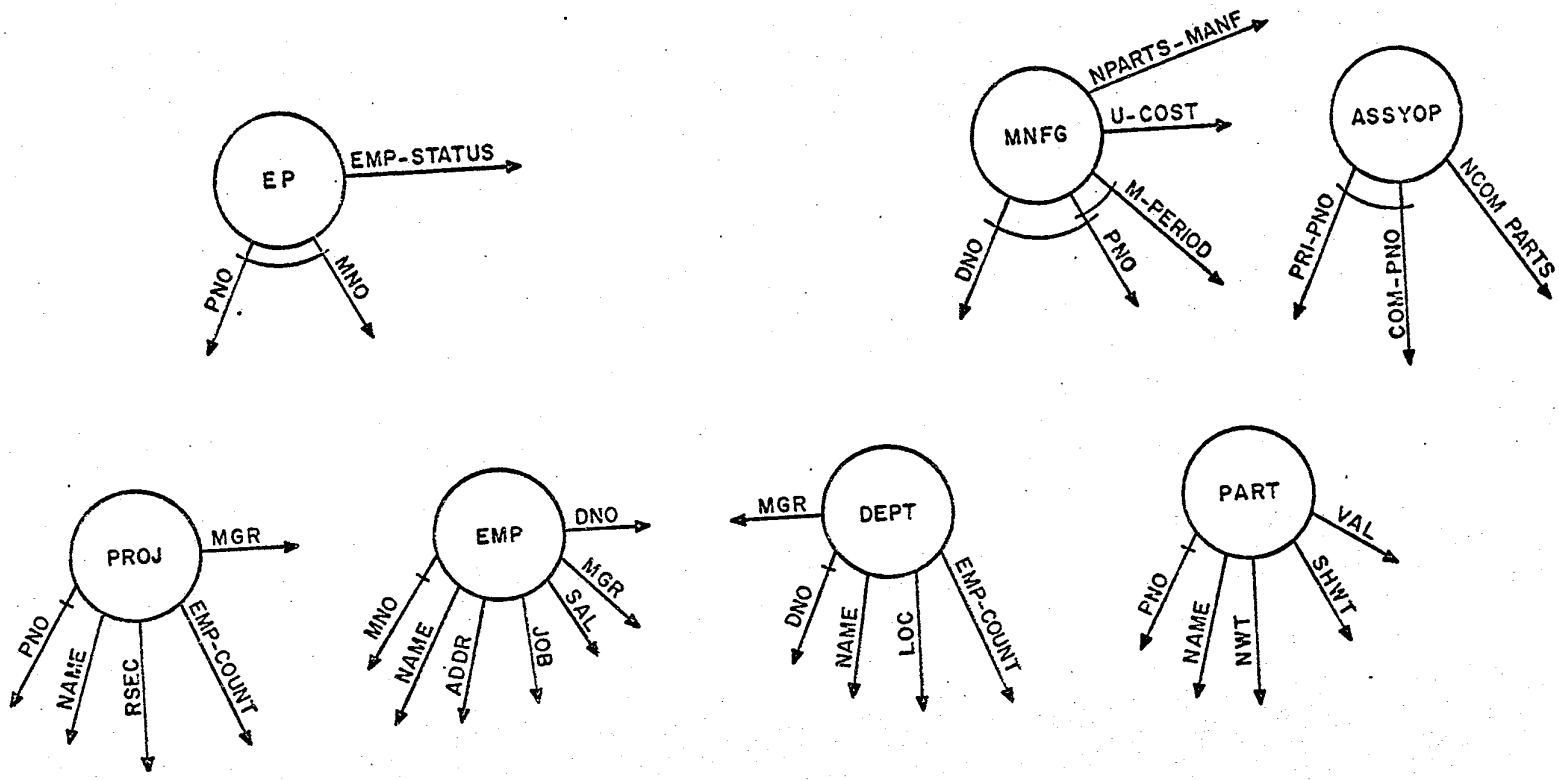
Finally, the Functional Model may be considered a link between the relational, DBTG/CODASYL, and entity set models, so that the Query Schema Syntax might serve as a common query language among them.

APPENDIX : Illustrations



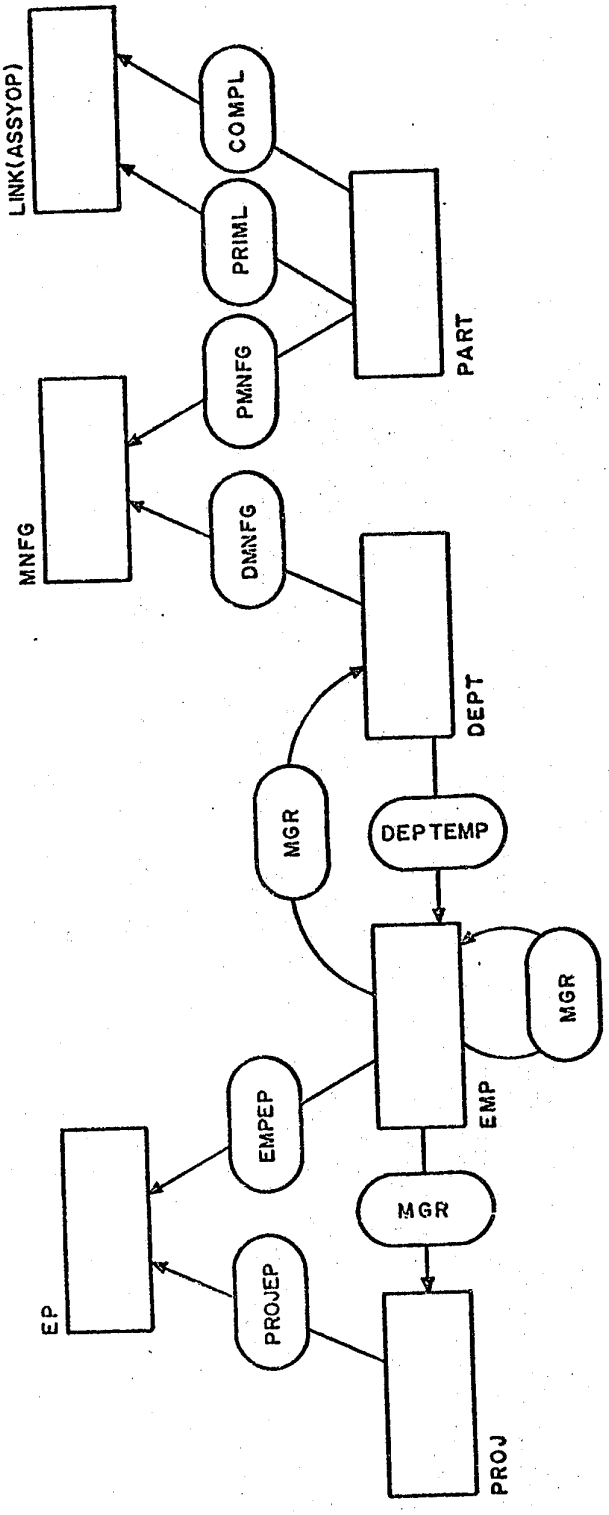
The Functional Model of a Firm

Figure 2



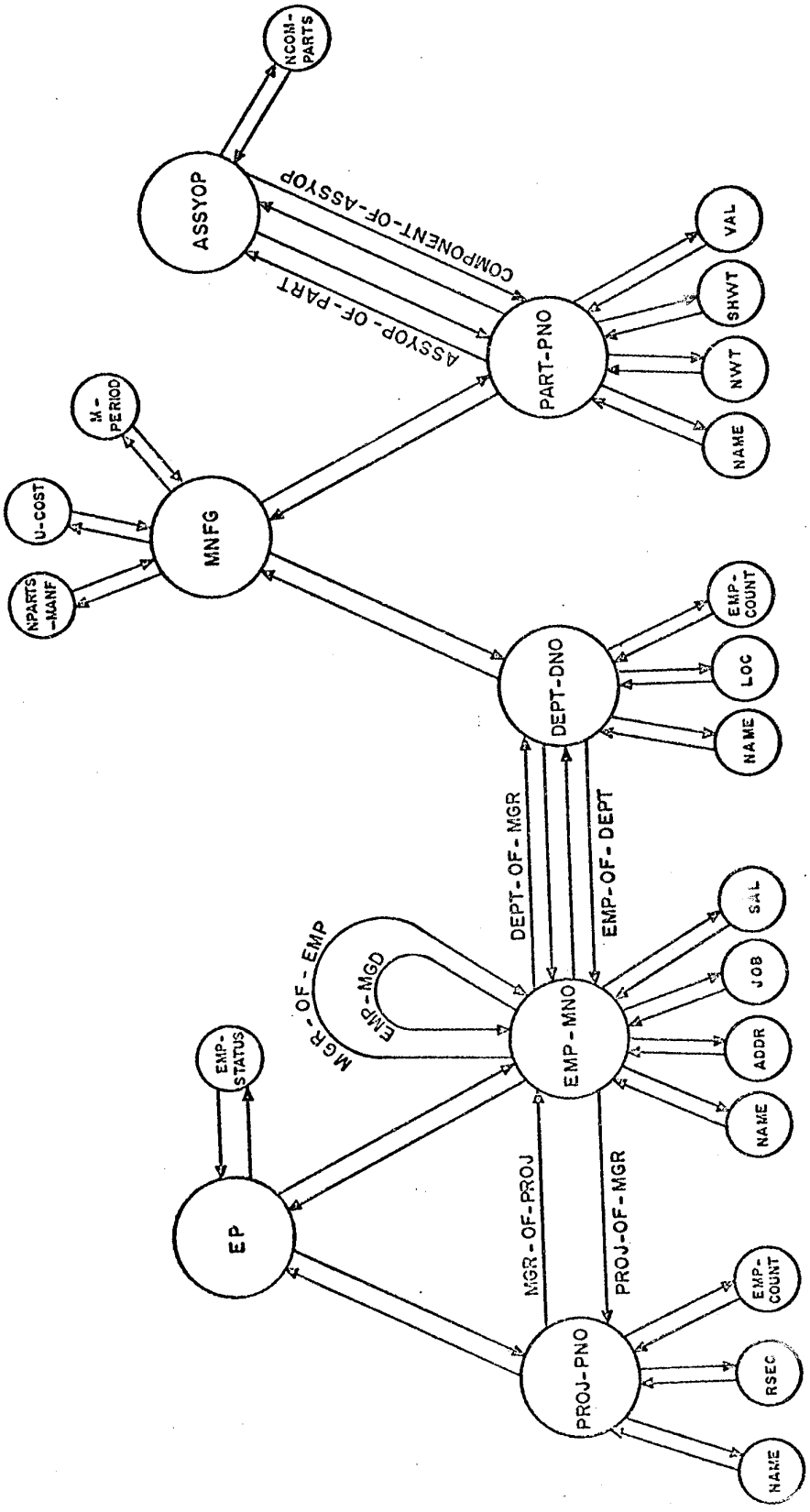
The Relational Model of the Firm

Figure 3



The DBTG / CODASYL Model of the Firm

Figure 4



The Entity Set / DIAM II Model of the Firm

Figure 5

REFERENCES

(1) CODASYL, 1971. CODASYL Data Base Task Group Report, April, 1971. ACM, NEW YORK.

(2) Senko, M.E., 1975. "The DDL in the Context of a Multilevel Structured Description: DIAM II with FOR AL", Data Base Description, North Holland.
Publ. Co., Amsterdam, 239-258.

(3) Cood, E.F., 1970. "A Relational Model of Data for Large Shared Data Banks", Communications ACM, vol. 13, no. 6, 377-387.

(4) Codd, E.F., 1971. "Further Normalization of the Data Base Relational Model", Courant Computer Science Symposia 6, Data Base Systems, Prentice-Hall.

(5) Wang, C.P. and Wedekind, H.H., 1975. "Segment Synthesis in Logical Data Base Design", IBM Journal Research and Development, vol.19, no. 1, 71-77.

(6) Schmid, H.A. and Swenson J.R., 1975. "On the Semantics of the Relational Data Model", ACM-SIGMOD, International Conference on Management of Data, May 14-16, 211-223.

(7) Nijssen, G.M., 1974. "Data Structuring in the DDL and Relational Model", Data Base Management, North-Holland Pub. Co., Amsterdam, 363-378.

(8) Tsichritzis, D. 1975. "A Network Framework for Relation Implementation", Computer System Research Group, University of Toronto.

(9) MacLane, S. and Birkhoff, G., 1967. Algebra, The Macmillan Co, New York.

(10) Harary, F., 1969. Graph Theory, Addison-Wesley Pub. Co., Reading, Massachusets.

(11) Codd, E.F., 1971. "A Data Base Sublanguage Founded on the Relational Calculus", Proceedings of 1971 ACM SIGFIDET Workshop on Data Description, Access and Control.

(12) Fehder, P., 1975. Personal Communication.

(13) Engeler, E., 1970. "Structure and Meaning of Elementary Programs", Symposium on Semantics of Algorithmic Languages, Lecture Notes in Mathematics 188, Springer-Verlag, New York.

(14) Kerschberg, L. and Pacheco, J.E.S., "The AQUERIUS Languages", to appear.

(15) Chamberlin, D.D. and Boyce, R.F., 1974. "SEQUEL: A Structured English Query Language", Proc. of 1974 ACM SIGFIDED Workshop, Ann Arbor, Michigan, April, 1974.

(16) Date, C.J., 1975. An Introduction to Database Systems, Addison-Wesley Publishing Co., Reading, Massachusetts.