数理解析研究所講究録
423巻 1981年 213-237

Correlating the Real World with One Relation Scheme Using the Functional Dependency Graph

Hirofumi Katsuno

Musashino Electrical Communication Laboratory, N.T.T.
3-9-11, Midori-cho, Musashino-shi
Tokyo, 180, Japan

This paper is a preliminary version. All the proofs are omitted. Most of them can be found in reference [8], however. The final version will be submitted for some journal.

1. Introduction

Database normalization theory for logically designing a relational database has been developed.[1] The theory studies algorithms for obtaining a set of relation schemes with good properties under the universal relation assumption. The input of the algorithms is a set of integrity constraints, known as data dependencies. Such dependencies are functional, multivalued, join dependencies, and so on.

When applying the algorithms to actual database design, three problems appear. The theory ignores how to derive other data dependencies than functional dependencies from the real world. Thorough study has not been made on how the universal relation assumption restricts modeling the real world as a relational database. The theory does not consider other important integrity constraints than data dependencies. That is, semantic aspects of database design have not been studied enough in database normalization theory, though syntactic aspects have been well studied.

Various data models, which describe the real world, have been proposed for database design.[3],[5] The models can handle semantic aspects of database design. It is not shown in the models, however, how other data dependencies than functional dependencies can be represented, and what is meant by the universal relation assumption.

This paper reports results of a study on semantic

aspects of database design and shows how to make it easier to apply database normalization theory to actual database design.

This paper presents a data model, called the functional dependency graph (FD-graph). The FD-graph clarifies which part of the real world can be represented as one relation scheme without null values. The FD-graph cannot represent the whole real world, however. That is, the FD-graph clarifies part of the real world which corresponds to a universal relation. The paper shows a method of constructing a relation scheme from an FD-graph. An FD-graph having no loop is discussed in [10].

The paper shows that multivalued dependencies in the relation constructed from an FD-graph correspond to certain FD-graph substructures. This result means that multivalued dependencies can be easily derived from the FD-graph.

The paper shows that an integrity constraint, specified in drawing an FD-graph and called domain equivalence conditions, is important in designing relational databases without null values.

The paper finally shows an algorithm for obtaining fourth normal form relation schemes by using an FD-graph.

The FD-graph is based on the functional dependency model (FD-model) developed by Housel, et al.[5] The FD-graph differs from the FD-model as follows. The FD-graph structure is rigidly defined, while that of the FD-model is not. The FD-model can handle the part of the real world

which should be represented as multiple relations, while the FD-graph handles only the part of the real world which can be represented as one relation. The FD-graph can include special nodes (join nodes), which represent a data dependency, such as an embedded multivalued dependency and which do not exist in the FD-model.

There are some similarities between the FD-graph and other data models, such as the entity-relationship model [3]. The results of this paper can be applied to those data models, since they can be easily transformed into a set of FD-graphs.

2. Basic Concepts

Let U be a set of attributes and be denoted by U={A1, A2,..., An}. Let each attribute Ai have its corresponding domain $\mathcal{D}$(Ai). A relation R(U) on U is a subset of $\mathcal{D}$(A1)X$\mathcal{D}$(A2)X...X$\mathcal{D}$(An), where X means a Cartesian product. An element of a relation is called a tuple of the relation.

A relation scheme is denoted by $\underline{R}$=<U,Γ>, where R is a relation name, U is an attribute set, and Γ is a set of its constraints. The relation scheme $\underline{R}$=<U,Γ> is defined as the set of relations on U which satisfy constraints Γ. An element of a relation scheme is called an instance of the relation scheme.

Let X be a subset of U and be denoted by X={B1, B2,...,

Bm}. For a tuple μ=<a1,a2,...,an>, the restriction of μ on X, denoted by μ[X], is defined as μ[X]=<b1,b2,...,bm>, where bi=aj if Bi=Aj. The projection of R(U) on X, denoted by π_X(R(U)), is defined as

π_X(R(U))={ μ[X] | $\mu \in$ R(U) }.

Let X, Y and Z be pairwise disjoint sets. The join of R(X,Y) and S(X,Z), denoted by R(X,Y)⋈S(X,Z), is defined as

R⋈S={ <x,y,z> | <x,y>∈R and <x,z>∈S }.

A functional dependency (FD) is a kind of constraint about R(U). Let X and Y be subsets of U. The FD: X-->Y holds in R(U), iff for every pair μ_1, μ_2 of tuples of R, if μ_1[X]=μ_2[X], then μ_1[Y]=μ_2[Y]. It is said that an FD: X-->Y is trivial if Y is a subset of X.

An embedded join dependency (EJD) is another constraint about R(U). Let X1, X2,..., and Xn be subsets of U. The EJD: ⋈{X1,X2,...,Xn} holds in R(U), iff for any tuples μ_1, μ_2,..., μ_n of R(U), if μ_i[Xi∩Xj]=μ_j[Xi∩Xj] (1≤i,j≤n) holds, there exists a tuple μ such that μ[Xi]=μ_i[Xi] (1≤i≤n). If $U=\bigcup_{i=1}^{n}$ Xi, ⋈{X1,X2,..,Xn} is called a join dependency (JD).

Let V be a subset of U. Let X and Y be subsets of V and let Z be defined by Z=V-(X∪Y). An embedded multivalued dependency (EMVD): X-->-->Y in V holds in R(U), iff ⋈{X∪Y,X∪Z} holds in R(U). If V=U, the dependency is called a multivalued dependency (MVD) and is denoted by X-->-->Y. It is said that an MVD: X-->-->Y is trivial if Y is a subset of X or U=X∪Y.

An FD (EJD,JD,EMVD,MVD) holds in $\underline{R}$=<U,Γ>, iff the FD

(EJD,JD,EMVD,MVD) holds in any instance of R.

If Γ is a set of data dependencies, R=<U,Γ> is the set of relations which satisfy all the data dependencies in Γ.

A relation scheme R=<U,Γ> is in Boyce-Codd normal form if, whenever a nontrivial FD: X-->Y holds in any instance of R, the FD: X-->U holds in any instance of R.

A relation scheme R=<U,Γ> is in fourth normal form if, whenever a nontrivial MVD: X-->-->Y holds in any instance of R, the FD: X-->U holds in any instance of R.

It is said that these normal forms have desirable properties.[3],[6] A different view about the desirableness is discussed in [9] under the domain equivalence condition, however.

Assume that a database is composed of R1(U1), R2(U2), ..., and Rn(Un). Then, the universal relation assumption means that there exists a relation R(U) such that $U=\bigcup_{i=1}^{n} Ui$ and $Ri(Ui)=\pi_{Ui}(R)$.

3. Functional Dependency Graph

The functional dependency graph (FD-graph) is defined in this section. The FD-graph is a data model whose role is to clarify which part of the real world can be represented by one relation without null values. The FD-graph is based on the functional dependency model developed by Housel et al.[5]

3.1 Functional Dependency Graph Definition

The structure of the FD-graph is defined and the basic meanings which an FD-graph represents in the real world is shown in this subsection. More precise meanings are described in 3.2.

Definition 1. A connected directed graph G is called a functional dependency graph candidate (FD-graph candidate), if G has the following properties:

(1) The nodes of G are divided into three categories: atomic nodes, compound nodes, and join nodes. The arcs of G are also divided into two categories: functional dependency arcs (FD-arcs) and constituent arcs (c-arcs).

(2) No atomic node has an outgoing c-arc.

(3) Any compound node or any join node must have multiple outgoing c-arcs. Each terminal node of the c-arcs is called a component of the compound (join) node.

(4) Each atomic node has its own specific label, which indicates an attribute name.

(5) If there is an arc from N to M, there is no other arc from N to M.

An example of such a graph is shown in Fig. 1. The node shown as a rectangle with a saltire cross is a join node. The nodes shown as a clear rectangle are compound nodes. The nodes shown as a rectangle with a noun are atomic nodes.

The noun is the label of the atomic node. That is, N1 is a join node; N2-N5 are compound nodes; The other nodes are atomic nodes; EXPERIMENT#, SCORE and so on are labels.

The arcs with a black arrow head are FD-arcs. The arcs with a clear arrow head are c-arcs. That is, f1 and f2 are FD-arcs; The other arcs are c-arcs.

An atomic node represents an entity indicated by its label. A compound node represents an abstract notion constructed from its components. The abstraction corresponds to aggregation in [6] and the abstracted notion is identified by a basic attribute set (see the following Definition 4). A compound node also represents an integrity constraint that no functional dependency holds among its components. That is, a compound node shows many-to-many relationship among its components.

A join node plays the same role as a compound node. In addition, a join node represents another integrity constraint that the relation which corresponds to data about the join node is the join of relations of its components (The precise definition is shown in Definition 6 (4)). For example, N1 shows that any JUDGE, who corresponds to a PERFUME, judges all the experiments concerned with the PERFUME.

An FD-arc shows that the basic attribute set of its terminal node is functionally dependent on that of its initial node. For example, CHARM-DEGREE is functionally dependent on {MODEL,JUDGE}.

Each node and each FD-arc represent the meanings just described. Various facts or data, which correspond to the meanings, exist in the real world. Restricting the facts or data makes it possible to represent more precise meanings for an FD-graph (see 3.2).

Definition 2. Let N and M be any nodes of an FD-graph candidate G. The FD-reachability from N to M, $N \xrightarrow[FD]{} M$, is inductively defined as follows:

(1) $N \xrightarrow[FD]{} N$

(2) If M is a compound (or join) node and $N \xrightarrow[FD]{} Mi$ for each compoent Mi ($1 \leq i \leq n$) of M, $N \xrightarrow[FD]{} M$ holds.

(3) If there is a node, L, such that $N \xrightarrow[FD]{} L$ and either $L \dashrightarrow M$ or $L \dashrightarrow M$ *), then $N \xrightarrow[FD]{} M$.

The component-reachablility from N to M, $N \xrightarrow[C]{} M$, is also defined as follows:

(1) $N \xrightarrow[C]{} N$

(2) $N \xrightarrow[C]{} L$ and $L \dashrightarrow M$ imply $N \xrightarrow[C]{} M$.

Example 1. $N1 \xrightarrow[FD]{} N13$. $N2 \xrightarrow[FD]{X} N13$.

Figure 2 is an FD-graph candidate. No compound node has an appropriate notion constructed from its components, however.

Figure 3 is another FD-graph candidate. The graph has a contradictory meaning. The compound node N means there is

*) $L \dashrightarrow M$ ($L \dashrightarrow M$) means that there exists an FD-arc (c-arc) from L to M.

a many-to-many relationship between PERSON and TELEPHONE#, where the FD-arc f means TELEPHONE# is uniquely determined by PERSON.

FD-graph candidates are confined to remove graphs such as Fig. 2 and Fig. 3.

Definition 3. If an FD-graph candidate G satisfies the following conditions, G is called a functional dependency graph (FD-graph).

(1) For any different nodes, N, M and L, if $N \xrightarrow{c} M$ and $N \xrightarrow{c} L$, $M \xrightarrow{FD} L$ does not hold.

(2) When we regard G as an undirected graph, if G has a primitive loop p, there exists a node N such that $N \xrightarrow{FD} M$ for any node M which is included in p.

(3) G has no directed loop composed of only c-arcs.

Condition (2) is necessary to represent facts or data on an FD-graph as one relation (see the proof of Theorem 1).

3.2 Functional Dependency Graph Instance

A collection of facts or data, which corresponds to the meanings of an FD-graph, is defined as an instance of the FD-graph. The "instance" notion gives an FD-graph more precise meanings in the real world.

Definition 4. For each node N of an FD-graph G, the basic attribute set X(N) is defined by

$X(N)=\{ A \mid N \xrightarrow[C]{} NA$, where NA is an atomic node whose label is attribute A$\}$.

Lemma 1. Let G be an FD-graph.

(1) $X(NA)=\{A\}$

(2) If a node N is either a compound node or a join node,

$X(N)=X(M1)\cup\ldots\cup X(Mk)$,

where M1,..., and Mk are all the components of N.

(3) For any node N, $X(N)\neq\phi$.

Example 2. X(N1)={MODEL, USED-AMOUNT, PERFUME, JUDGE}. X(N4)={MODEL, JUDGE}.

Definition 5. For each node N, let $\mathcal{N}$ be defined as

$\mathcal{N}=\{ M \mid N \xrightarrow[FD]{} M \}$.

The derived attribute set Y(N) is defined as

$Y(N)=\bigcup_{M\in\mathcal{N}} X(M)$.

The derived attribute set of N indicates all the entities functionally determined by the notion which corresponds to N.

Example 3. $Y(N1)=X(N1)\cup$ {EXPERIMENT#, SCORE, CHARM-DEGREE, CONTENT}.

Definition 6. Let N1, N2,..., Nn be all the nodes of an FD-graph G. Let f1, f2,..., fm be all the FD-arcs of G. Instance candidate I of G is defined as

I={R(N1,I),..., R(Nn,I), fI1,..., fIm}, where

(1) R(N,I) is a relation on X(N). The relation is called the basic relation of N.

(2) If $X(N) \cap X(M) = X \neq \phi$,

$\pi_X[R(N,I)] = \pi_X[R(M,I)]$.

For any c-arc c: N--▷M, let cI be $\pi_{X(M)}$. It follows from condition (2) that cI is a surjective function from R(N,I) to R(M,I).

(3) For any FD-arc f: N-→M, fI is a surjective function from R(N,I) to R(M,I).

(4) For any join node N,

R(N,I)=R(M1,I)⋈...⋈R(Mk,I)

where M1,..., Mk are all the components of N.

Basic relation R(N,I) is a set of facts or data about the many-to-many relationship concerned with N. Function fI is a set of facts or data about the functional relationship indicated by FD-arc f.

Conditions (2) and (3) demand that a function which corresponds to each arc is surjective. The surjectivity is very important, when an instance of G is represented as one relation without null values. The conditions about the surjectivity are called domain equivalence conditions.

There is a case where Condition (2) is not satisfied.

Let us consider an FD-graph G in Fig.4 and an instance candidate I of G. Let R(N1,I) be the data which show what kind of hobbies each person has and R(N2,I) be the data which show what kind of hobbies each person wants to have in the future. The set I={R(N1,I),R(N2,I)} is not an instance candidate of G, since I does not satisfy Condition (2).

It follows from Condition (2) and (4) that a join dependency $\bowtie$\{X(M1),...,X(Mk)\} holds in R(N,I) for each join node N, where M1,...,Mk are all the components of N.

Definition 7. An instance candidate I of G is called an instance of G, if I satisfies the following condition.

For any pair of nodes <N,M>, $N \xrightarrow[FD]{} M$, there exists a function FNM such that

(1) FNM is a surjective function from R(N,I) to R(M,I).

(2) If there exists an arc from N to M, FNM=aI, where aI is the function which corresponds to the arc and is determined by either condition (2) or (3) in Definition 6.

(3) If $N \xrightarrow[FD]{} L$ and $L \xrightarrow[FD]{} M$, FNM=FLMFNL holds.

There is an instance candidate which is not an instance. Let G be an FD-graph shown in Fig.5. Let function fI1 represent each employee's manager. Let function fI2 represent each manager's sex. Let function fI3 represent each employee's sex. Then, the instance candidate I, constructed from fI1, fI2, and fI3, is not an instance of G,

since Condition (3) does not hold.

Lemma. Let I be an instance candidate of an FD-graph G. If G has no loop when G is regarded as an undirected graph, I is an instance of G.

3.3 Relational Representation of Functional Dependency Graph Instance

This subsection shows that data for an instance can be represented by one relation without loss of information. Domain equivalence condition and Condition (2) in Definition 3 are the key to the representation by one relation.

Lemma. Let I be an instance of an FD-graph G.

(1) Let p=(al,a2,...,an) be a directed path from N to M, where each ai is an arc of G, the terminal node of ai is the initial node of ai+1, the initial node of al is N, and the terminal node of an is M. Then,

FNM=aInaIn-1...aIl,

where aIi is the function determined by the arc ai as shown in Definition 6.

(2) If $M \underset{C}{\dashrightarrow} L$, $FML=\pi_{X(L)}$.

Definition 8. For an FD-graph G and an instance I of G, the derived relation S(N,I) on Y(N) is defined as follows:

(1) If X(N)=Y(N), S(N,I)=R(N,I).

(2) If X(N)=Y(N), let {A1,...,Ak} be Y(N)-X(N). Let Mi be the atomic node whose label is Ai. Then,

$S(N,I)=\{\mu \mid \mu[X(N)]\in R(N,I)$ and $\mu[Ai]=FNMi(\mu[X(N)])$ $(1\leq i\leq k)\}$

Lemma. Let I be an instance of an FD-graph G.

(1) $\pi_{X(N)}(S(N,I))=R(N,I)$

(2) X(N) is a key for S(N,I).

(3) If an FD-arc f: N-→M exists, an FD: X(N)-->X(M) holds in S(N,I). When the FD is regarded as a function, the function is exactly fI.

(4) If $N \underset{FD}{\dashrightarrow} M$, $\pi_{Y(M)}(S(N,I))=S(M,I)$.

Definition 9. A node N is called a maximal node, if N satisfies one of the following conditions for any node M.

(1) $M \underset{FD}{\not\rightarrow} N$

(2) $M \underset{FD}{\dashrightarrow} N$ and $N \underset{FD}{\dashrightarrow} M$

Definition 10. Let N1, N2,..., and Nk be all the maximal nodes of an FD-graph G. The relational representation of I is defined by

$$R(G,I)=\bigbowtie_{i=1}^{k} S(Ni,I).$$

Theorem 1.

(1) For any node N, $\pi_{Y(N)}(R(G,I))=S(N,I)$.

(2) If an FD-arc f: N-→M exists, an FD: X(N)-->X(M) holds in R(G,I). When the FD is regarded as a function, the

fuction is exactly fI.

Theorem 1 shows that the relational representation of I inherits all the information from I without loss. Theorem 1 does not hold, if domain equivalence condition or Condition (2) in Definitin 3 does not hold.

Definition 11. A relation scheme $\underline{R}_G=\langle U,\Gamma_G\rangle$ is called the relation scheme of an FD-graph G, if for any instance of $\underline{R}_G$, $R_G(U)$, there exists an instance of G, I, such that $R(G,I)=R_G(U)$.

4. Data Dependencies in Relation Scheme of Functional Dependency Graph

This section clarifies which data dependencies characterize the relation scheme of an FD-graph G and shows that multivalued dependencies can be derived from certain substructures of G.

Definition 12. Let $\mathcal{N}$ be $\mathcal{N}=\{ M \mid N \xrightarrow[FD]{} M\}$. Let G' be the graph made from G by removing all the nodes in $\mathcal{N}$ and all the arcs whose initial or terminal node is included in $\mathcal{N}$. When G' is regarded as an undirected graph, each connected component Hi of G' is called an associated collection with N. Let Wi be the set of attributes which are the label of

an atomic node in Hi. Wi is called an associated attribute set of Hi with N.

Theorem 2. Let N be any node of an FD-graph G and Wi be any associated attribute set with N. Then,

X(N)-->-->Wi

holds in the relation scheme $\underline{R}_G=\langle U,\Gamma_G\rangle$ of G.

When G has no join node, let W' be a nonempty proper subset of Wi. Then,

X(N)-->-->W'

does not hold in $\underline{R}_G$.

Definition 13. Let G be an FD-graph.

(1) The set of FDs, Γ_1^G, is defined as

Γ_1^G={X(N)-->X(M) | An FD-arc: N-▶M exists in G},

and is called the essential functional dependency set of G.

(2) The set of MVDs, Γ_2^G, is defined as

Γ_2^G={X(N)-->-->Wi | Wi is an associated attribute set with N and Y(N)$^{\cup}$Wi$\neq$U},

and is called the essential multivalued dependency set of G.

(3) The set of EJDs, Γ_3^G, is defined as

Γ_3^G={ γ | γ is an embedded join dependency indicated by a join node},

and is called the essential embedded join dependency set of G.

(4) The essential data dependency set of G, Γ^G, is defined by

$$\Gamma^G = \Gamma_1^G \cup \Gamma_2^G \cup \Gamma_3^G .$$

Example 4. Let G be the FD-graph shown in Fig. 1. All the elements in Γ_2^G are:

{MODEL,USED-AMOUNT,PERFUME}-->-->{INGREDIENT,CONTENT}|
{EXPERIMENT#,SCORE,CHARM-DEGREE,JUDGE},

{PERFUME,JUDGE}-->-->{INGREDIENT,CONTENT}|
{EXPERIMENT#,SCORE,MODEL,USED-AMOUNT,CHARM-DEGREE},

{PERFUME}-->-->{INGREDIENT,CONTENT}|
{EXPERIMENT#,SCORE,MODEL,USED-AMOUNT,JUDGE,CHARM-DEGREE}.

Γ_3^G is composed of an EMVD:

{PERFUME}-->-->{MODEL,USED-AMOUNT}|{JUDGE} in
{MODEL,USED-AMOUNT,PERFUME,JUDGE}.

Theorem 3. Let G be an FD-graph. If any data dependency of Γ^G holds in a relation R(U), R(U) is an instance of the relation scheme $\underline{R}_G$=<U,Γ_G> of G.

Corollary 1. $\underline{R}_G$=<U,Γ_G> is equivalent to $\underline{R}^G$=<U,Γ^G> in the sense that each relation scheme has the same set of instances.

Corollary 1 means that the relation scheme of G is characterized by the essential dependency set of G. The result makes it possible to normalize a relation scheme based on an FD-graph instead of a set of data dependencies. One of such attempts is shown in the next section.

5. Algorithm for obtaining fourth normal form relation schemes

An algorithm for obtaining fourth normal form relation schemes is shown in this section.

Definition 15. Let G be an FD-graph and N be a maximal node of G. Let $\mathcal{N}$ and $\mathcal{M}$ be defined by $\mathcal{N}=\{ M \mid N \xrightarrow[FD]{} M\}$ and $\mathcal{M}=\{ M \mid M \in \mathcal{N}$ and there exists a directed path from another maximal node to M}, respectively. The N-graph GN is defined as the subgraph of G constructed from all the nodes of $\mathcal{N}$ and arcs between them. The N-delete-graph FN is defined as the subgraph of G obtained by removing all the nodes of $\mathcal{N}-\mathcal{M}$.

Lemma. The N-graph GN is an FD-graph and each connected component of the N-delete-graph FN is also an FD-graph.

Algorithm

Input: G, where G is an FD-graph with no join node.

Output: $\mathcal{S}$, where $\mathcal{S}$ is a set of relation schemes.

1. Let Max={N1, N2,..., Nn} be the set of all the maximal nodes of G. Let Max'={M1, M2,..., Mm} be a subset of Max such that

   (1) $Y(Mi) \neq Y(Mj)$ if $i \neq j$,

   (2) For any Ni, there exists Mj, where Y(Ni)=Y(Mj).

2. Let G1 be the M1-graph of G. Let F1 be the M1-delete-graph of G. Let H2 be the set of FD-graphs which

are a connected component of F1.

3. i:=2

4. If i>m, then go to 7, else H:=Hi. (H is a set of FD-graphs.)

5. Let G' be the FD-graph which is an element of H and has Mi as a maximal node. Let Gi be the Mi-graph of G'. Let G'' be the Mi-delete-graph. Hi+1:=Hi$^{\cup}${G''}-{G}.

6. i:=i+1; go to 4.

7. Let $\underline{R}$i=<Ui,Γi> be the relation scheme of Gi. Then, Γi includes no MVD.

8. Let $\mathcal{S}$i be the set of Boyce-Codd normal form relation schemes which is obtained from Ri by the algorithm in [2]. Let $\mathcal{S}$ be defined as $\mathcal{S}=\bigcup_{i=1}^{m}\mathcal{S}_i$.

The algorithm may not produce fourth normal form relation schemes, since algorithms which always produce Boyce-Codd normal form relation schemes do not exist.

Theorem 4. Let R be any instance of $\underline{R}_G$=<U,Γ_G>. Let $\mathcal{S}$ be denoted by $\mathcal{S}$={ $\underline{S}$i=<Vi,Φi> | i=1,2,...,k }. Let Si be defined by Si=π_{V_i}(R). Then,

(1) Si is an instance of Si=<Ui,Φi>.

(2) R=S1$\bowtie$S2$\bowtie$...$\bowtie$Sm

(3) Let Γ_1^G be the essential functional dependency set of G. $\Gamma_1^{G+}=(\bigcup_{i=1}^{m}\Phi i)^{+}$, where + means the closure.

(4) Φ_i includes no MVD.

6. Concluding Remarks

This paper clarifies which part of the real world can be represented as one relation scheme by using a data model, called the functional dependency graph (FD-graph). The paper shows that multivalued dependencies, which hold in the relation scheme constructed from an FD-graph, can be easily derived from certain substructures of the FD-graph.

The domain equivalence condition which appears in the definition of an instance of the FD-graph is important in designing relational databases. The method of designing relational databases by using FD-graphs is discussed in [9].

Acknowledgement

The author thanks the members of the first research section for their advice and encouragement.

References

[1] C. Beeri, P.A. Bernstein and N. Goodman: A Sophisticate's Introduction to Database Normalization Theory, Proc 4th Inter. Conf. on VLDB, pp.113-124, 1978

[2] J. Biskup, U. Dayal and P.A. Bernstein: Synthesizing Independent Database Schemas, Proc. 1979 ACM-SIGMOD,

pp.143-151

[3] P. Chen:The Entity-Relationship Model--Toward a Unified View of Data, ACM Trans. on Database Syst., Vol.1, No.1, pp.9-36, 1976

[4] R. Fagin: Normal Forms and Relational Database Operators, Proc. 1979 ACM-SIGMOD, pp.153-160

[5] B.H. Housel, V. Waddle and S.B. Yao: The Functional Dependency Model For Logical Design, Proc. 5th Inter. Conf. on VLDB, pp.194-208, 1979

[6] J.M. Smith and D.C.P. Smith: Database Abstractions: Aggregation, CACM, Vol.20, pp.405-413, 1977

[7] J.D. Ullman: Principles of Database Systems, Computer Science Press, 1980

[8] H. Katsuno: Relationship Between Functional Dependency Graph and Data Dependencies in Relational Databases, AL80-25, 1980 (in Japanese)

[9] H. Katsuno: An Application of Database Normalization Theory to Database Design, AL80-52, 1980 (in Japanese)

[10] H. Katsuno: On Two Different Meanings of Multivalued Dependencies in a Conceptual Schema, to appear in J. IECE, Japan

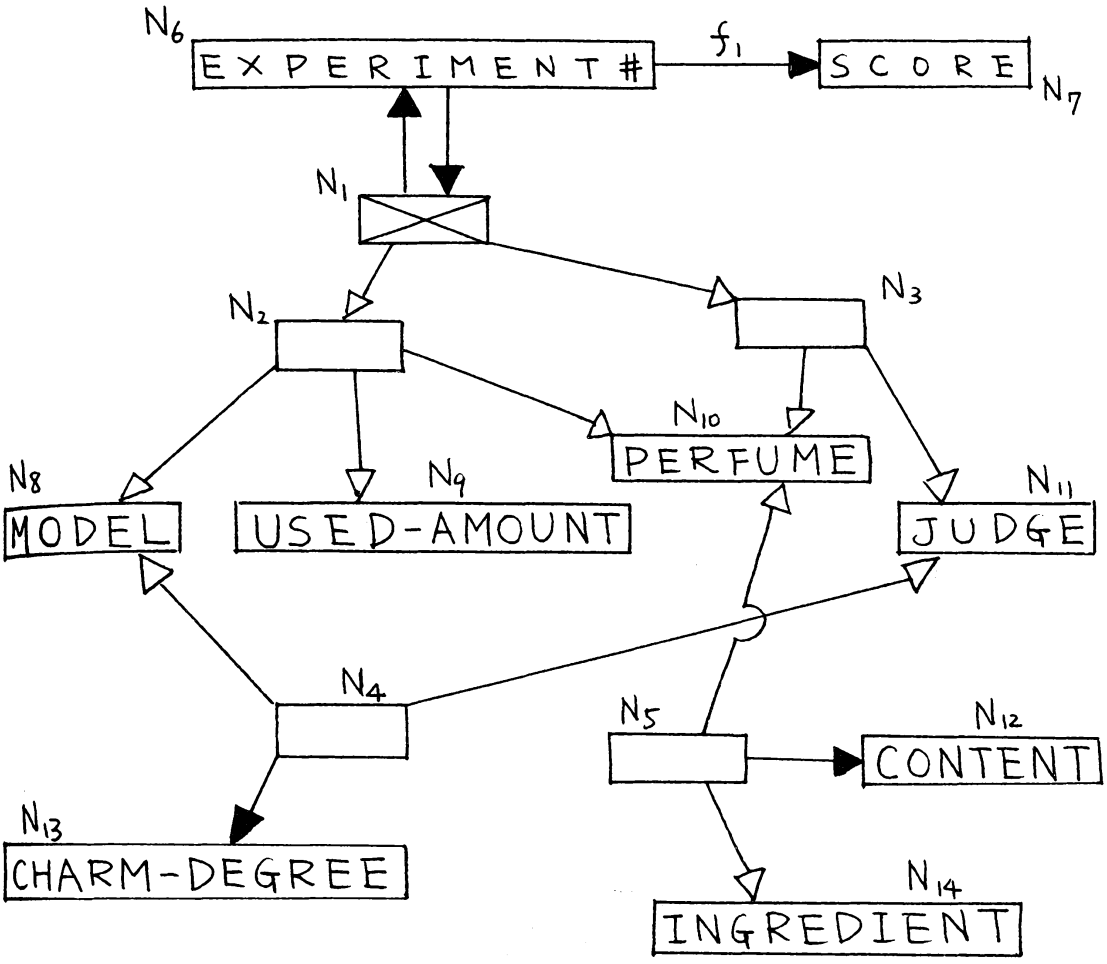

Figure 1 A functional dependency graph for perfume experiment

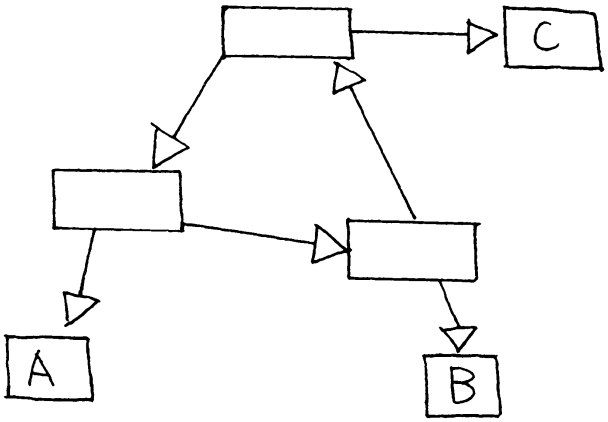

Figure 2 A functional dependency graph candidate

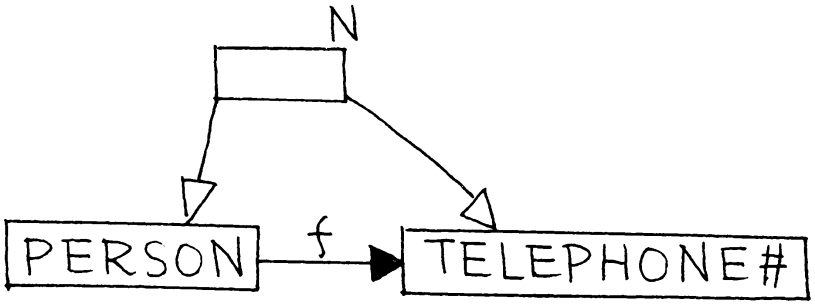

Figure 3 Telephone functional dependency graph candidate

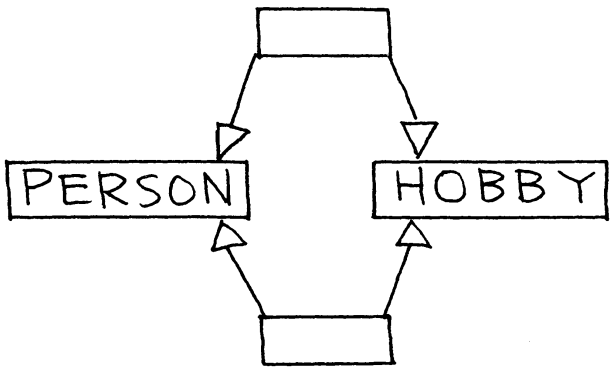

Figure 4 A functional dependency graph

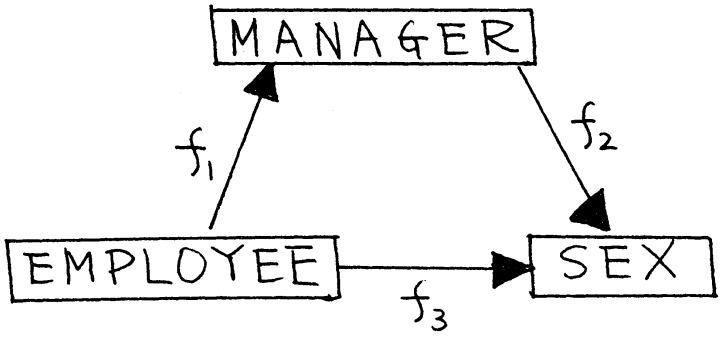

Figure 5 Manager functional dependency graph