IMPLICIT CONSTRAINTS WITHIN RELATIONAL DATA BASES

Tok-Wang Ling
Frank Tompa

Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada

CS-78-35
July 1978

This research reported here was supported in part by grants from the National Research Council of Canada.

ABSTRACT

In this paper, we show that a relational data base which consists of a set of relations may contain some implicit constraints which are induced by functional dependencies. These constraints, in turn, induce updating dependencies among the relations. To discover all implicit constraints, an algebraic representation for describing the functional dependencies in the closure of a given set of functional dependencies is presented. Conditions for a functional dependency to induce a constraint in the *minimum* set of implicit constraints are given. As a result, given an optimal set of relations, its minimum set of implicit constraints can be derived by using the algebraic representations of the functional dependencies involved.

Key words and phrases: data base design, relational data bases, optimal sets of relations, implicit constraints, label expressions.

CR Categories: 3.70, 4.33

0. Introduction

The problem of maintaining the integrity of a data base is to ensure that the stored data is accurate at all times. Integrity can be achieved, to a certain extent, by protecting and preventing the stored data against invalid updates of the data base due to carelessness or lack of knowledge of the users. Different ways of classifying integrity constraints are described in several articles [Date 77, Hammer 75, Kapali 75, Furtado 77, Stonebraker 75]. All these integrity constraints must be given by the data base designers and described in the conceptual schema by some language, e.g. predicate calculus notation or any other language of equivalent expressive power such as SEQUEL [Chamberlin 74].

It is not our aim to study all these types of integrity constraint. Rather, in this paper we examine some hidden integrity constraints called implicit constraints within relational data bases which contain more than one relation. This type of constraint is due to the functional dependencies of a set of relations, and occurrences are easily overlooked by data base designers (section 2). In order to discover all these implicit constraints in relational data bases, we give an algebraic representation for describing a functional dependency in the closure of a given set of functional dependencies (section 3). This algebraic representation explicitly records in a single expression, all ways of deriving a functional dependency and from this, we can derive the implicit constraints. Results concerning the minimum set of implicit constraints for a given optimal set of relations are given in section 3.3.

1. The relational model

A relational data base, consisting of several interrelated relations, was first introduced by Codd [Codd 70]. A relation is defined as follows:

Given sets of atomic (non-decomposable) elements $D_1, D_2, \ldots, D_n$ (not necessarily distinct), R is a first normal form relation (or simply relation) on these n sets if it is a set of ordered n-tuples $(d_1, d_2, \ldots, d_n)$ such that d_i belongs to D_i, for $i = 1,2,\ldots,n$, i.e. $R \subseteq D_1 \times D_2 \times \ldots \times D_n$, where $D_1 \times D_2 \times \ldots \times D_n$ denotes the Cartesian product of D_1, D_2, ..., and D_n. We call D_1, D_2, ..., D_n the domains of R. Rather than referencing each domain by a position number, we give each one a unique role name called an attribute of R and refer to the elements of the domain having attribute name b as b-values. We say that a set of attributes B of R is functionally dependent on a set of attributes A of R iff each A-value in R has associated with it at most one B-value in R (at any time). We denote this by $A \rightarrow B$ and call it a functional dependency of R. A functional dependency $A \rightarrow b$ of R, where A is a set of attributes of R and b is a single attribute of R, is said to be a full dependency of R (or b is fully dependent on A) iff there exists no proper subset A_1 of A such that $A_1 \rightarrow b$. A set of attributes K is called a candidate key (or simply key) of R iff all attributes of R are functionally dependent on K, and no proper subset of K has this property. It is easy to prove that every relation has at least one key, and some relations may have more than one key. When a relation has more than one key, it is customary to designate one of the keys as the

primary key of the relation. To avoid identification problems, all attributes in the primary key must have defined (i.e. non-null) values for each triple [Ling 78b].

We denote keys of a relation by underlining the attributes, and the primary key by a double underline if the relation has more than one key. For example $R_1(\underline{a,b},c,d)$ has one key, namely {a,b} which is also the primary key of R_1, whereas the relation $R_2(\underline{\underline{a}},\underline{b},\underline{c,d},e,f)$ has three keys, namely {a}, {b} and {c,d}, and {a} is the primary key of R_2. Because single letters are used for attributes names, in this case we can simply denote {a}, {b} and {c,d} by a, b and cd respectively without causing any ambiguity.

Given a set of functional dependencies F over a set of attributes A, we define the closure of F, denoted by F^+, as follows [Osborn 78]:

(1) $F \subseteq F^+$,

(2) projectivity: for all subsets X and Y of A, if $Y \subseteq X$ then $X \rightarrow Y \in F^+$,

(3) transitivity: for all subsets X, Y and Z of A, if $X \rightarrow Y$, $Y \rightarrow Z \in F^+$ then $X \rightarrow Z \in F^+$,

(4) additivity: for all subsets X, Y and Z of A, if $X \rightarrow Y$, $X \rightarrow Z \in F^+$ then $X \rightarrow Y \cup Z \in F^+$ (or simply denoted by $X \rightarrow YZ$),

(5) no other functional dependencies are in F^+.

Because of the property of additivity, we can assume a canonical form, in which for each functional dependency exactly one attribute appears on the right hand side. Given a relation R, we project it into a set of relations $\mathcal{R} = \{R_1, R_2, \ldots, R_m\}$. Two distinct relations R_i and R_j of $\mathcal{R}$ have the non-loss property iff for any instance I of R, if we replace I by its projections onto the R_i's then we can recover the projection of I onto the union of the attributes of R_i and R_j by using the join of the instances of R_i and R_j (a formal definition is given in [Aho 77]). Aho has shown that two relations R_i and R_j have a non-loss join iff $R_i \cap R_j \rightarrow R_i$ or $R_i \cap R_j \rightarrow R_j$, where R_i and R_j here denote the attributes of the relations R_i and R_j respectively [Aho 77]. The set $\mathcal{R}$ is said to be reconstructable if we can derive the original relation R without any loss of information [Osborn 78].

Given a set of relations $\mathcal{R}$ and a set of functional dependencies G defined on $\mathcal{R}$. $\mathcal{R}$ is said to be optimal if and only if for any two different relations R_1 and R_2 of $\mathcal{R}$ with keys K_1 and K_2 respectively, $K_1 \rightarrow K_2 \notin G^+$ or $K_2 \rightarrow K_1 \notin G^+$.

Given an optimal set of relations $\mathcal{R}$ and a set of functional dependencies G defined on $\mathcal{R}$. A key K of a relation R of $\mathcal{R}$ is called an explicit key of the relation R if there exists an attribute b in R and $b \notin K$ such $K \rightarrow b \in G$. A key of a relation R of $\mathcal{R}$ which is not an explicit key is called an implicit key of R [Ling 78a].

2. Implicit Constraints in a Relational Data Base

Let $\mathcal{R} = \{R_1, R_2, \ldots, R_n\}$ be an optimal set of relations and the functional dependency set of $\mathcal{R}$ be $F(\mathcal{R})$:

$$F(\mathcal{R}) = \{K \to b \mid K \text{ is an explicit key of some relation } R \text{ of } \mathcal{R},\ b \text{ is an attribute of } R \text{ and } b \notin K\}.$$

We say that $\mathcal{R}$ covers a set of functional dependencies G if and only if $F^+(\mathcal{R}) = G^+$. Let $FD(R_i)$ represent the restriction of $F^+(\mathcal{R})$ to the relation R_i, that is,

$$FD(R_i) = \{A \to b \in F^+(\mathcal{R}) \mid A \cup \{b\} \text{ are attributes of } R_i\}$$

for all $i = 1,2,\ldots,n$. Then $\bigcup_i FD(R_i) \subseteq F^+(\mathcal{R})$ and, if we also require covering during normalization, $(\bigcup_i FD(R_i))^+ = F^+(\mathcal{R})$.

Now let us consider the updating of a relation $R_i \in \mathcal{R}$. Under what conditions does an updated data base instance still satisfy $F^+(\mathcal{R})$? Ensuring that the updated instance of the relation R_i satisfies all functional dependencies in $FD(R_i)$ does not, in general, ensure that the updated data base instance will satisfy $F^+(\mathcal{R})$, as we can see from the following examples:

Example 1. Let $\mathcal{R} = \{R_1(\underline{a},b,c), R_2(\underline{b},d), R_3(\underline{c},d)\}$ be **an optimal set of relations. Clearly**
$F(\mathcal{R}) = \{a \to b, a \to c, b \to d, c \to d\}$, $FD(R_1) = \{a \to b, a \to c\}^+$, $FD(R_2) = \{b \to d\}^+$ and $FD(R_3) = \{c \to d\}^+$. An instance of this data base $\mathcal{R}$ is shown in figure 1.

a	b	c
a_1	b_1	c_1
a_2	b_2	c_2

$R_1(\underline{a},b,c)$

b	d
b_1	d_1
b_2	d_2

$R_2(\underline{b},d)$

c	d
c_1	d_1
c_2	d_2
c_3	d_1

$R_3(\underline{c},d)$

Figure 1. An instance of the data base.

Clearly this data base instance satisfies all functional dependencies in $F^+(R)$. Consider updating R_2 by changing the tuple (b_1,d_1) to (b_1,d_2). The updated instance of R_2 as shown in Figure 2 satisfies $FD(R_2)$.

a	b	c
a_1	b_1	c_1
a_2	b_2	c_2

$R_1(\underline{a},b,c)$

b	d
b_1	d_2
b_2	d_2

$R_2(\underline{b},d)$

c	d
c_1	d_1
c_2	d_2
c_3	d_1

$R_3(\underline{c},d)$

Figure 2. The instance of the data base after updating.

Considering $F^+(R)$, since $a \rightarrow b$, $b \rightarrow d \in F(R)$, therefore $a \rightarrow d \in F^+(R)$. But we see that $a \rightarrow d$ is not true in the updated instance of R, since $(a_1,b_1,c_1) \in R_1$ and $(b_1,d_2) \in R_2$, so the a-value a_1 is associated with the d-value d_2; also $(a_1,b_1,c_1) \in R_1$ and $(c_1,d_1) \in R_3$, so the a-value a_1 is associated with the d-value d_1, i.e. the a-value a_1 is associated with two d-values d_1 and d_2. It is also important to notice that even though this updated data base instance does not satisfy $F^+(R)$, it still does satisfy all functional dependencies in $F(R)$.

One thing we should notice is that if $A \rightarrow b$ is a functional dependency in $F^+(R)$ and A is not contained in any relation in R, then by the non-loss property, the existence of an A-value implies that this A-value can be obtained by projecting a tuple which belongs to a non-loss join of some relations of R. For example, the natural join of the instances of R_2 and R_3 in figure 1 is $\{(b_1,c_1,c_1), (b_1,c_3,d_1), (b_2,c_2,d_2)\}$. However, this join is lossy: only b_1c_1 and b_2c_2 are existing bc-values of the data base instance of R, but b_1c_3 is not.

Let us look at another example.

Example 2. Let $R = \{R_1(\underline{a,b},c), R_2(\underline{c},b)\}$ be an optimal set of relations. If we want to update an instance of R_2, we should check whether the updated instance has no conflict with the relationships between b and c in the instance of R_1 and the instance of R_2.

From the above discussion, we have the following conclusion:

Assume an optimal set of relations R and the functional dependencies set F(R) of R. It is not true that an instance of the data base which satisfies all functional dependencies in F(R) will automatically satisfy all functional dependencies in $F^+(R)$; also when we update a relation R_i of R, it is not always sufficient to check only whether the updated instance of R_i still satisfies the functional dependencies in $FD(R_i)$.

This means that we need to find the conditions (i.e. constraints) for update validity. We call this kind of constraint implicit functional dependency constraints (or simply implicit constraints) within a set of relations. A formal definition will be given in section 3.2.

3. Algebraic Representation of the Functional Dependencies in a Closure

Given that $A \rightarrow B$ is in G^+, the closure of a given set of functional dependencies G, we know that $A \rightarrow B$ can be derived from G. However, which functional dependencies and which rules are involved when deriving $A \rightarrow B$ are usually not explicitly recorded. Adaptations of derivations and derivation trees from formal languages to express how functional dependencies are derived in G^+ [Bernstein 75], record only one derivation even when there may be several. Here we give an algebraic approach in which all ways of deriving a functional dependency in G^+ can be captured by a single expression. By using these expressions we can find all implicit constraints within a relational data base.

3.1. Basic Mathematical Background

To date, there is little mathematical basis for manipulating functional dependencies described in the literature. The concepts of partial functions capture the properties of functional dependencies. Thus in this section, we give a basic partial function theory which is used for describing implicit constraints within relational data bases.

Let $A = \{A_1, A_2, \ldots, A_n\}$ and $B = \{B_1, B_2, \ldots, B_n\}$ be two non-empty collections of non-empty sets. For simplicity, we will write

$f : A \to B$, $a \in A$, $b \in B$, $(a,b) \in f$, $(a,b) \in A \times B$ and $f(a) = b$, by which we mean

$$f : A_1 \times A_2 \times \ldots \times A_n \to B_1 \times B_2 \times \ldots \times B_m;$$

$$a = (a_1, a_2, \ldots, a_n) \in A_1 \times A_2 \times \ldots \times A_n, \quad a_i \in A_i;$$

$$b = (b_1, b_2, \ldots, b_m) \in B_1 \times B_2 \times \ldots \times B_m, \quad b_j \in B_j;$$

$$(a_1, \ldots, a_n, b_1, \ldots, b_m) \in f;$$

$$(a_1, \ldots, a_n, b_1, \ldots, b_m) \in A_1 \times \ldots \times A_n \times B_1 \times \ldots \times B_m;$$

and $f(a_1, \ldots, a_n) = (b_1, \ldots, b_m)$ respectively.

A <u>relationship</u>* f from A to B is a subset of $A \times B$. A <u>partial function</u> f from A to B, denoted by $f : A \to B$, is a relationship from A to B such that for every $a \in A$, there exists at most one $b \in B$ such that $(a,b) \in f$. If $(a,b) \in f$, then we write $f(a) = b$.

Let A, B and C be three non-empty collections of non-empty sets. Let $f : A \to B$ and $g : B \to C$ be two partial functions. The composition of f and g, denoted by $f \circ g$, is defined as follows:

$$f \circ g = \{(a,c) \mid a \in A, c \in C, \exists\, b \in B \text{ such that } (a,b) \in f \text{ and } (b,c) \in g\}.$$

---

*Note that this term is called a relation in set theory, but that would lead to confusion here.

It is obvious that the composition $f \circ g$ is still a partial function and is defined from A to C [Stanat 77].

Let $f : A \to B$ and $g : A \to C$ be two partial functions and $B \cap C = \phi$. The <u>additive join</u> of f and g, denoted by $f \wedge g$, is defined as follows:

$$f \wedge g = \{(a,b,c) \mid a \in A,\ b \in B,\ c \in C \text{ such that } (a,b) \in f \text{ and } (a,c) \in g\}.$$

It is clear that the additive join $f \wedge g$ is also a partial function and is defined from A to $B \times C$.

Let $f : A \to B$ and $g : A \to B$ be two partial functions defined on the same domains. The <u>union</u> of f and g, denoted by $f + g$, is defined as follows:

$$f + g = \{(a,b) \mid a \in A,\ b \in B \text{ such that } (a,b) \in f \text{ or } (a,b) \in g\}.$$

It is not difficult to show that the union $f + g$ defines a relationship between A and B which is not necessarily a partial function, as we can see from the following example:

Let $A = \{1,2,3\}$, $B = \{a,b,c\}$.

Let $f : A \to B$ with $f = \{(1,a), (2,b)\}$ and

$g : A \to B$ with $g = \{(2,a), (3,c)\}$.

We have $f + g = \{(1,a), (2,a), (2,b), (3,c)\}$, and $f + g$ is not a partial function from A to B since $(2,a)$ and $(2,b)$ are in $f + g$.

Two partial functions f and g defined from A to B are equal (or identical) if

$$(a,b) \in f \quad \text{iff} \quad (a,b) \in g \quad \text{for all} \quad a \in A, \quad b \in B.$$

We denote this by $f = g$. We say that f and g agree iff $(a,b) \in f$ and $(a,b') \in g$ implies $b = b'$ for all $a \in A$, $b \in B$ and $b' \in B$ [Bourbaki 68].

It is clear that if two partial functions are equal then they also agree, but the converse is not true, as we can see from the following example:

Let $A = \{1,2,3\}$, $B = \{a,b,c\}$.

Let $f : A \to B$ and $g : A \to B$ and $f = \{(2,b), (3,c)\}$ and $g = \{(1,a), (2,b)\}$.

By the definition, f and g agree, but clearly they are not equal: e.g., f(3) is defined but g(3) is not.

Lemma 1. Let $f : A \to B$ and $g : A \to B$ be two partial functions from A to B. The union of f and g, i.e. $f + g$, is a partial function from A to B iff f and g agree.

Let $f : A \to B$ and $g : C \to D$ be two partial functions. We say $f \theta g$ is defined where $\theta \in \{\circ,+,\wedge\}$ iff

(1) $\theta = \circ$ and $B = C$, or

(2) $\theta = +$, $A = C$ and $B = D$, or

(3) $\theta = \wedge$ and $A = C$.

Let α, β and γ be three partial functions. In the following

properties, the equality is true iff the right hand side and left hand side are defined.

P1. (Idempotence of +.) $\alpha + \alpha = \alpha$

P2. (Commutativity of +.) $\alpha + \beta = \beta + \alpha$

P3. (Commutativity of $\wedge$.) $\alpha \wedge \beta = \beta \wedge \alpha$

P4. (Associativity of +.) $(\alpha + \beta) + \gamma = \alpha + (\beta + \gamma)$

Hence we can simply denote $(\alpha + \beta) + \gamma$ or $\alpha + (\beta + \gamma)$ by $\alpha + \beta + \gamma$ without any ambiguity.

P5. (Associativity of $\wedge$.) $(\alpha \wedge \beta) \wedge \gamma = \alpha \wedge (\beta \wedge \gamma)$

Hence we can simply denote $(\alpha \wedge \beta) \wedge \gamma$ or $\alpha \wedge (\beta \wedge \gamma)$ by $\alpha \wedge \beta \wedge \gamma$ without any ambiguity.

P6. (Associativity of $\circ$.) $(\alpha \circ \beta) \circ \gamma = \alpha \circ (\beta \circ \gamma)$

Hence we can simply denote $(\alpha \circ \beta) \circ \gamma$ or $\alpha \circ (\beta \circ \gamma)$ by $\alpha \circ \beta \circ \gamma$ without any ambiguity.

P7. (Left distributivity of $\wedge$ over +.) $\alpha \wedge (\beta + \gamma) = (\alpha \wedge \beta) + (\alpha \wedge \gamma)$

P8. (Left distributivity of $\circ$ over +.) $\alpha \circ (\beta + \gamma) = (\alpha \circ \beta) + (\alpha \circ \gamma)$

P9. (Left distributivity of $\circ$ over $\wedge$.) $\alpha \circ (\beta \wedge \alpha) = (\alpha \circ \beta) \wedge (\alpha \circ \beta)$

P10. (Right distributivity of $\circ$ over +.) $(\beta + \gamma) \circ \alpha = (\beta \circ \gamma) + (\gamma \circ \alpha)$

For convenience, we give priorities to these operators by giving $\circ$ the highest priority and $+$ the lowest priority. Hence $(\alpha \wedge \beta) + (\alpha \wedge \gamma)$, $(\alpha \circ \beta) + (\alpha \circ \gamma)$ and $(\alpha \circ \beta) \wedge (\alpha \circ \gamma)$ can be written as $\alpha \wedge \beta + \alpha \wedge \gamma$, $\alpha \circ \beta + \alpha \circ \gamma$ and $\alpha \circ \beta \wedge \alpha \circ \gamma$, respectively.

3.2. Label Expression of a Functional Dependency

In this section, we will give an algebraic representation for the functional dependencies in the closure of a given set of functional dependencies. We first define what is meant by a partial label of a functional dependency in the closure of a given set of functional dependencies.

Definition 1. Let G be a set of functional dependencies which is in canonical form and consists of k functional dependencies named by $1,2,\ldots,k$ respectively. Let A, B and C be three non-empty sets of attributes of G. The partial labels of functional dependencies of G^+ are defined recursively as follows:

(1) For each $i \in \{1,2,\ldots,k\}$, i is a partial label of the functional dependency in G with name i.

(2) (Projectivity.) If $B \subseteq A$, then P_A^B is a partial label of the functional dependency $A \to B \in G^+$.

(3) (Additivity.) If α and β are partial labels of functional dependencies $A \to B$ and $A \to C$ of G^+ respectively and $B \cap C = \phi$, then $\alpha \wedge \beta$ is a partial label of $A \to BC \in G^+$.

(4) (Transitivity.) If α and β are partial labels of functional dependencies $A \to B$ and $B \to C$ of G^+ respectively, then $\alpha \circ \beta$ is a partial label of $A \to C \in G^+$.

(5) (Alternativity.) If α and β are partial labels of a functional dependency $A \to B \in G^+$, then $\alpha + \beta$ is also a partial label of $A \to B \in G^+$.

(6) No other expressions are partial labels of functional dependencies of G^+.

The semantics of partial labels can be described as follows:

Assume an optimal set of relations R with the set of functional dependencies $G = F(R)$ which is in canonical form and consists of k functional dependencies named by $1,2,\ldots,k$.

(1) For each $i \in \{1,2,\ldots,k\}$, a partial label i of the functional dependency $A \to b$ with name i of G, at any time t, defines a partial function

$$f_t : \mathcal{D}(a_1) \times \ldots \times \mathcal{D}(a_\ell) \to \mathcal{D}(b)$$

where $A = \{a_1,\ldots,a_\ell\}$ and $\mathcal{D}(a_i)$ denotes the domain of the attribute a_i. This partial function f_t is obtained by projecting the relation R of R, in which A is a key of R and b is an attribute of R, on the attributes set $A \cup b$. For simplicity, we also denote f_t by $f_t : A \to b$ and if there is no ambiguity, we simply denote f_t by f.

(2) Let A and B be two non-empty sets of attributes such that $B \subseteq A$ and A be a key of some relation R of R. P_A^B is a partial label of $A \to B \in G^+$ and at any time t, defines a partial function $f : A \to B$. This partial function f is obtained by projecting the relation R on A, and for all $a \in A$ in the projection, $f(a)$ is defined as the projection of a on the attribute set B. Note that we do not consider the case when A is not a key of any relation in R, since we do not need this type of projection (see section 3.3).

(3) Let A, B and C be three non-empty sets of attributes of G and $B \cap C = \phi$. Let α and β be partial labels of $A \to B$ and $A \to C$ of G^+ respectively. If at time t, α and β define

relationships f (from A to B) and g (from A to C) respectively, then the partial label $\alpha \wedge \beta$, which is a partial label of $A \rightarrow BC \in G^+$, at time t defines the additive join $f \wedge g$ of f and g.

(4) Let A, B and C be three non-empty sets of attributes of G, and let α and β be partial labels of $A \rightarrow B$ and $B \rightarrow C$ of G^+ respectively. If at time t, α and β define relationships f (from A to B) and g (from B to C) respectively, then the partial label $\alpha \circ \beta$, which is a partial label of $A \rightarrow C \in G^+$, at time t defines the composition $f \circ g$ of f and g.

(5) Let A and B be two non-empty sets of attributes of G. Let α and β be partial labels of $A \rightarrow B \in G^+$. If at time t, α and β define relationship f and g (from A to B) respectively, then the partial label $\alpha + \beta$, which is also a partial label of $A \rightarrow B \in G^+$, at time t defines the union $f + g$ of f and g.

All the above interpretations can be proved to be valid by the definitions of partial label and closure of a given set of functional dependencies. Now we can define what is meant by a claim that two partial labels of a functional dependency are equal.

Definition 2. Let α and β be two partial labels of a functional dependency $A \rightarrow B \in F^+(R)$, where R is an optimal set of relations. We say that α and β are equal, denoted by $\alpha = \beta$, if and only if α and β define the same relationship from A to B at all times.

It is easy to show the partial labels of $F^+(R)$ satisfy the properties P1 to P10 given in section 3.1. We have the following additional properties:

P11. If $B \subseteq A$, $C \subseteq A$, $B \cap C = \phi$ then $P_A^B \wedge P_A^C = P_A^{BC}$.

P12. $P_A^A \circ \alpha = \alpha \circ P_B^B = \alpha$.

P13. If $C \subseteq B \subseteq A$ then $P_A^B \circ P_B^C = P_A^C$.

Given an optimal set of relations R, let α be a partial label of a functional dependency in $F^+(R)$. By the properties of partial labels (P7, P8 and P9), it is clear that α can be expressed in disjunctive normal form [Stanat 77] i.e.

$$\alpha = \alpha_1 + \alpha_2 + \ldots + \alpha_n, \quad n \geq 1,$$

and each α_i does not contain "+" , for all $i = 1,2,\ldots,n$. We call each α_i a term of α.

Definition 3. Let $\alpha = \alpha_1 + \alpha_2 + \ldots + \alpha_n$, $\alpha \geq 1$ be a partial label and in disjunctive normal form. α is irreducible iff $\alpha_i \neq \alpha_j$ for $i \neq j$, $i, j = 1,2,\ldots,n$.

Lemma 2. Let R be an optimal set of relations and A and B be two non-empty sets of attributes of R. We have:

(1) If $\alpha = \alpha_1 + \alpha_2 + \ldots + \alpha_n$ $(n \geq 1)$ is a partial label of a functional dependency $A \rightarrow B \in F^+(R)$ and is in disjunctive normal form, then each term α_i of α defines a way to derive $A \rightarrow B$ in $F^+(R)$.

(2) For any functional dependency $A \rightarrow B \in F^+(R)$, there exists a partial label $\alpha = \alpha_1 + \ldots + \alpha_n$ $(n \geq 1)$ which is in disjunctive normal form, such that for any way of deriving $A \rightarrow B \in F^+(R)$

there exists a $j \in \{1,2,\ldots,n\}$ such that α_j represents this way of deriving $A \to B \in F^+(R)$.

We say α in (2) of lemma 2 is complete.

Corollary 1. Any two complete and irreducible partial labels of a functional dependency in $F^+(R)$ have the same number of terms and they are equal, i.e. they define the same relationship for attributes which appear in the functional dependency.

From now on, we call a complete and irreducible partial label of a functional dependency a label of the functional dependency. Note that if we give some method for ordering labels of functional dependencies, then each functional dependency in $F^+(R)$ can be labelled by **a unique label which has the minimum rank in that ordering among labels of the functional dependency.**

Let us look at some examples:

Example 3. Let $R = \{R_1(\underline{a},b,c), R_2(\underline{b},d), R_3(\underline{c,d},f)\}$ be an optimal set of relations. Clearly $F(R) = \{a \to b, a \to c, b \to d, cd \to f\}$. Let us name the functional dependencies in $F(R)$ by 1, 2, 3 and 4. We see that 1 and 3 are partial labels of $a \to b$ and $b \to d$ respectively, hence $1 \circ 3$ is a partial label of $a \to d \in F^+(R)$. Also 2 is a partial label of $a \to c$, so $1 \circ 3 \wedge 2$ (i.e. $(1 \circ 3) \wedge 2$) is a partial label of $a \to cd \in F^+(R)$. Now $1 \circ 3 \wedge 2$ and 4 are partial labels of $a \to cd$ and $cd \to f$ of $F^+(R)$ respectively, so $(1 \circ 3 \wedge 2) \circ 4$ is a partial label of $a \to f \in F^+(R)$. Note that $(2 \wedge 1 \circ 3) \circ 4$ is also a partial label of $a \to f \in F^+(R)$, **but we can** show that it is equal to $(1 \circ 3 \wedge 2) \circ 4$ and both are complete and

irreducible, so both are labels of $a \to f \in F^+(R)$.

Example 4. Let $R = \{R_1(\underline{a},b,c), R_2(\underline{b},d), R_3(\underline{c},d)\}$ be an optimal set of relations. Clearly $F(R) = \{a \to b, a \to c, b \to d, c \to d\}$. Let us name the functional dependencies in $F(R)$ by 1, 2, 3 and 4. Since 1 and 3 are partial labels of $a \to b$ and $b \to d$ respectively, hence $1 \circ 3$ is a partial label of $a \to d \in F^+(R)$. Similarly, 2 and 4 are partial labels of $a \to c$, $c \to d$ respectively, hence $2 \circ 4$ is a partial label of $a \to d \in F^+(R)$. Thus $1 \circ 3 + 2 \circ 4$ is also a partial label of $a \to d \in F^+(R)$, and we can show that it is complete and irreducible, hence it is a label of $a \to d \in F^+(R)$. Note that neither $1 \circ 3$ nor $2 \circ 4$ is complete.

We now give a formal definition for implicit constraint.

Definition 4. Let R be an optimal set of relations and $F(R)$ be the set of functional dependencies of R. We say that R contains an implicit constraint induced by a functional dependency $A \to b \in F^+(R)$ iff a disjunctive normal form label of $A \to b$ has the form

$$\alpha = \alpha_1 + \alpha_2 + \ldots + \alpha_n, \quad n \geq 2.$$

That constraint is: at any time, the partial functions defined by each α_i, $i = 1,2,\ldots,n$ must agree.

Thus any time we want to update a relationship involving a functional dependency in $F(R)$, and this functional dependency appears in α, then we may have to check whether all partial functions defined by the α_i agree after updating. If not we should either reject the updating request or do some other updating to ensure that all partial

functions defined by α_i agree.

3.3. Minimum Implicit Constraints of a Relational Data Base

It may be that a disjunctive normal form label $\alpha = \alpha_1 + \alpha_1 + \ldots + \alpha_n$ $(n \geq 2)$ is "induced" by another implicit constraint β in the sense that if the updated data base instance satisfies β then it necessarily satisfies α. Thus it is possible that we need not check whether or not the updated data base instance satisfies the implicit constraint α every time we update a relationship of a functional dependency in $F(R)$ which is contained in α. Let us show this by the following example:

Example 5. Let $R = \{R_1(\underline{a},b,c), R_2(\underline{b},d), R_3(\underline{c},d), R_4(\underline{d},e)\}$ be an optimal set of relations.
Clearly $F(R) = \{a \to b, a \to c, b \to d, c \to d, d \to e\}$. Let us name the functional dependencies in $F(R)$ by 1, 2, 3, 4 and 5. It can be shown that $a \to d$ and $a \to e$ are in $F^+(R)$ with labels $\beta = 1 \circ 3 + 2 \circ 4$ and $\alpha = 1 \circ 3 \circ 5 + 2 \circ 4 \circ 5$ respectively. We can rewrite α as $(1 \circ 3 + 2 \circ 4) \circ 5$ (by P10). If a data base instance satisfies the implicit constraint β, then it also automatically satisfies the implicit constraint α.

We have the following result:

Lemma 3. Let R be an optimal set of relations and $F(R)$ be the set of functional dependencies of R. Let α be a label of a functional dependency $A \to B \in F^+(R)$, where A and B are non-empty sets of attributes of R. If $\alpha = \beta \ \theta \ \gamma$, where β and γ are

partial labels of some functional dependencies of $F^+(R)$ and θ is either $\wedge$ or $\circ$, then we can ignore α as an implicit constraint of R.

Proof. If $f : A \to B$, $g : A \to C$ and $h : B \to D$ are three partial functions, then $f \wedge g$ and $f \circ h$ are also partial functions from A to $B \times C$ and from A to D respectively.

Now let $\alpha = \beta \theta \gamma$ be a label of some functional dependency of $F^+(R)$ and θ be either $\wedge$ or $\circ$. If a data base instance satisfies the implicit constraints induced by β and γ, then β and γ define two partial functions, say f_1 and f_2. By the results stated in section 3.1, $f_1 \theta f_2$ is also a partial function, which implies that the data base instance also satisfies $\beta \theta \gamma$, i.e. α. This proves the lemma.

□

Theorem 1. Let R be an optimal set of relations with the functional dependencies set $F(R)$. To ensure that all implicity constraints are satisfied, we need to consider only the implicit constraints induced by functional dependencies in $F^+(R)$ which have the form $K \to b$, where K is a primary key of some relation in R and b is an attribute of R.

Proof. From lemma 3, we can ignore the implicit constraints induced by the functional dependencies of $F^+(R)$ which have the form

$$A \to B \in F^+(R) \quad \text{with} \quad |B| \geq 2$$

since these have labels of the form $\beta \wedge \gamma$. Suppose that $A \to b \in F^+(R)$ and A is not a primary key of any relation of R. Assume that there exists an A-value a_1 and two b-values b_1 and b_2 such that a_1 is associated with b_1 and b_2.

Case (1). If there exists a relation R of $\mathcal{R}$ with primary key K, and a tuple t of the relation instance of R such that $t[A]^{*} = a_1$, then $t[K]$ is defined and is associated with a_1, since all primary key values of tuples are defined. Hence $t[K]$ is associated with b_1 and b_2. Since $K \rightarrow A$, $A \rightarrow b \in F^{+}(\mathcal{R})$, therefore $K \rightarrow b \in F^{+}(\mathcal{R})$. By assumption, the data base instance satisfies $K \rightarrow b$, hence $b_1 = b_2$.

Case (2). If there exists no relation instance with a tuple t such that $t[A] = a_1$ then the existence of the A-value a_1 implies that a_1 can be constructed from a set of relations $\{R_1, R_2, \ldots, R_n\}$ and A is contained in the union of all attributes in these relations. By the property of reconstructability and the non-loss property there exists a $j \in \{1,2,\ldots,n\}$ and a tuple t of the instance of R_j such that $R_j \rightarrow R_1 \circ R_2 \circ \ldots \circ R_n$ and $t[K]$ is associated with a_1, where K is the primary key of R_j. Hence $t[K]$ is associated with b_1 and b_2. Since $K \rightarrow A$, $A \rightarrow b \in F^{+}(\mathcal{R})$ therefore $K \rightarrow b \in F^{+}(\mathcal{R})$. By assumption, the data base instance satisfies $K \rightarrow b$, therefore $b_1 = b_2$. Thus we can ignore the implicit constraint induced by $A \rightarrow b \in F^{+}(\mathcal{R})$ if A is not a primary key of any relation in $\mathcal{R}$. This proves the theorem. □

---

*$t[A]$ denotes the projection **t** on the attribute set A.

4. Accommodating Implicit Constraints

A tool called the functional dependency table [Ling 78b], which is an extension of an implication matrix [Fadous 75], can be used to find the minimum set of implicit constraints within a given relational data base. Briefly, given an optimal set of relations, R, the functional dependency table is used to derive the labels of elements in $F^+(R)$ for which the left hand sides constitute the primary keys of some relations in R and the right hand sides are single attributes. A complete description is given in [Ling 78b], where it is shown that the algorithm involved runs in time polynomial in the number of functional dependencies and the number of attributes.

4.1. Incidence of Implicit Constraints

Given a set of relations R, if there exist two relations R_1 and R_2 with keys K_1 and K_2 which are properly equivalent, that is $K_1 \rightarrow K_2 \in F^+(R)$ and $K_2 \rightarrow K_1 \in F^+(R)$, then we can merge R_1 and R_2 into one relation R such that R has an attribute set equal to the union of the attribute sets of R_1 and R_2, and the keys of R are all the keys of R_1 and R_2. Denote this by $R = \text{merge}(R_1, R_2)$.

Lemma 4: Let R be a non-optimal set of relations. Suppose there exist two relations R_1 and R_2 of R with primary keys K_1 and K_2 which are properly equivalent. Let $R = \text{merge}(R_1, R_2)$ and let $R' = R - \{R_1, R_2\} \cup \{R\}$. If there exist two disjoint non-empty sets of attributes A and B of relations R_1 and R_2 respectively with $B \cap \text{attribute}(R_1) = \phi$, and an attribute $d \in K_1$, such that $AB \rightarrow d$ is a full dependency in $F^+(R)$, we

have: R' has simpler implicit constraints than R in the sense that fewer relations need to be involved in implicit constraints in R'.

Proof. Let us first consider the case where K_1 is also a key of R_2. It can be shown that there exists an implicit constraint in R which can be expressed as

$$\beta = ((K_1 \rightarrow A) \wedge (K_1 \rightarrow B)) \circ \alpha + P^d_{k_1}$$

where α is a label expression for the functional dependency $AB \rightarrow d \in F^+(R)$. The relations which are involved in β are R_1, R_2 and the set of relations which are involved in α. Similarly, in R', there exists an implicit constraint which also can be expressed as β, and relations involved in β are R and the relations which are involved in α, hence the number of relations involved in β of R' is one less than in β of R.

Now let us consider the case in which all keys of R_2 are not keys of R_1. It can be shown that there exists an implicit constraint in R which can be expressed as

$$\beta_1 = ((K_1 \rightarrow A) \wedge (\alpha_1 \circ (K_2 \rightarrow B))) \circ \alpha_2 + P^d_{k_1}$$

where α_1 and α_2 are label expressions for functional dependencies $K_1 \rightarrow K_2$ and $AB \rightarrow d$ in $F^+(R)$ respectively. The relations which are involved in β_1 are R_1, R_2 and relations which are involved in α_1 and α_2. Similarly, there exists an implicit constraint in R' which can be expressed as

$$\beta_2 = ((K_1 \rightarrow A) \wedge (K_1 \rightarrow B)) \circ \alpha_2 + P^{d}_{k_1}.$$

Hence the relations which are involved in β_2 are R and relations which are involved in α_2, and therefore the number of relations involved in β_2 in R' is less than in β_1 in R. This proves the lemma. □

Example 6 Let $R = \{R_1(\underline{a_1,b_1},b_2,c), R_2(\underline{a_2,b_2},b_1,d), R_3(\underline{c,d},b_1), R_4(\underline{a_1},\underline{a_2})\}$. It can be shown that a_1b_1 and a_2b_2 are properly equivalent. R has an implicit constraint which can be represented by

$$\beta = ((a_1b_1 \rightarrow c) \wedge (((a_1b_1 \rightarrow b_2) \wedge (P^{a_1}_{a_1b_1} \circ (a_1 \rightarrow a_2))) \circ (a_2b_2 \rightarrow d))) \circ (cd \rightarrow b_1) + P^{b_1}_{a_1b_1}.$$

All four relations are involved in β. Now if we merge R_1 and R_2 into R, we get

$$R' = \{R(\underline{a_1,b_1,a_2,b_2},c,d), R_2(\underline{c,d},b_1), R_4(\underline{a_1},\underline{a_2})\}.$$

The implicit constraint in R' becomes

$$\beta_1 = (a_1b_1 \rightarrow cd) \circ (cd \rightarrow b_1) + P^{b_1}_{a_1b_1}$$

and

$$\beta_2 = (a_1b_1 \rightarrow a_2) + P^{a_1}_{a_1b_1} \circ (a_1 \rightarrow a_2).$$

The relations involved in β_1 and β_2 are R, R_3 and R_4, which is one less than in β.

Boyce-Codd form relations [Codd 74] are sometimes thought to contain no functional dependency constraints other than the key constraints, and hence they are thought to be better than non-Boyce-Codd form relations in the sense that they have less constraints to be maintained. This is not necessarily true when we consider a set of relations, as we can see from the following examples:

Example 7 Let $R_1 = \{R_1(\underline{a},\underline{b_1,b_2}), R_2(\underline{k,a},b_1,c)\}$ and $R_2 = \{R_1(\underline{a},\underline{b_1,b_2}), R_2'(\underline{k,b_1},b_2,c)\}$. We can show that $F^+(R_1) = F^+(R_2)$. We see that R_2 is a set of Boyce-Codd form relations and there is no implicit constraint in R_2. R_1 is not a set of Boyce-Codd form relations and there exists an implicit constraint which is represented by

$$\beta = (ka \rightarrow b_1) + (P^a_{ka} \circ (a \rightarrow b_1)).$$

Hence R_2 is better than R_1.

Let us look at another example which was given by Osborn [Osborn 78].

Example 8 Let $R_1 = \{R_1(\underline{a,b,c}), R_2(\underline{b,c,d},e), R_3(\underline{e},c)\}$ and $R_2 = \{R_1(\underline{a,b,c}), R_2'(\underline{a,d},e), R_3(\underline{e},c)\}$. We see that $F^+(R_1) = F^+(R_2)$, R_2 is a set of Boyce-Codd form relations but R_1 is not. In R_1 there is an implicit constraint which is represented by

$$\beta_1 = (bcd \rightarrow e) \circ (e \rightarrow c) + P^c_{bcd}$$

and the relations involved are R_2 and R_3. In R_2 there is also an implicit constraint which is represented by

$$\beta_2 = (ad \rightarrow e) \circ (e \rightarrow c) + (P^a_{ad} \circ (a \rightarrow c))$$

and the relations involved are R_1, R_2^+ and R_3. It is clear that to maintain the implicit constraint β_1 in R_1 is easier than to maintain the implicit constraint β_2 in R_2 in the sense that β_2 involves more relations. This example shows that some non-Boyce-Codd form relations sets are better than Boyce-Codd form relations for maintaining the implicit constraints.

From the above two examples, we see that although Boyce-Codd form relations set is sometimes better than non-Boyce-Codd form relations set, the reverse may just as well be true.

4.2. Checking implicit constraints

As described so far, implicit constraints are expressed in terms of functional dependencies rather than in terms of the relations involved. It may be better to express the implicit constraints in terms of relations by using relational algebra expressions [Codd 71], since relational data bases are conventionally described in terms of operations on relations rather than functional dependencies. We now define what is meant by restricted natural join of two relations, and then give a method to transform label expressions into relational algebra expressions.

Let R_1 and R_2 be two relations and A be a key of R_1 and $A \subseteq R_2$ (i.e. $A \subseteq$ attribute(R_2)). The restricted natural join of R_1 and R_2 on A, denoted by $(R_1 \circ R_2)\{A\}$, is defined as follows:

(1) If $R_1 \cap R_2 = A$ then, the restricted natural join of R_1 and R_2 on A is the same as the natural join of R_1 and R_2 [Codd 71], and we simply denote it by $R_1 \circ R_2$.

(2) If $R_1 \cap R_2 \supseteq A$, then the restricted natural join of R_1 and R_2 on A is the same as the natural join of R_1 and the relation obtained from R_2 by assigning different new names (but remember the original names) for those attributes of R_2 which belong to $R_1 \cap R_2 - A$. If there is no ambiguity, we denote the join by $R_1 \circ R_2$.

Note that each new name in the join represents an implicit constraint: the projections of every tuple of the join on this new name and the original name of the attribute must agree. Note that the restricted natural join of two relations is non-loss whereas the natural join of two relations may be lossy (as was described in section 2).

Now we informally present a method to transform a label expression α of a functional dependency $A \rightarrow b$ into a relational algebra expression.

> First express the label expression α in disjunctive normal form. Then delete any term that only contains a single projective function. For each other term, find all relations having functional dependencies involved in the term,

and project the relations on the attributes which are involved in the term. Then find the restricted natural join of these projected relations and project the join on $A \cup \{b\}$ and the new attributes.

Let us look at the following example:

Example 9 Let $\mathcal{R} = \{R_1(\underline{a,b},c), R_2(\underline{c},g)\}$. An implicit constraint of $\mathcal{R}$ is $P^b_{ab} + (ab \to c) \circ (c \to b)$. To transform this to a relational algebra expression, we drop the term P^b_{ab}. $(ab \to c) \circ (c \to b)$ involves R_1 and R_2, the join key is c, and b belongs to both relations but not a key, hence we rename the attribute b of R_2 by b', and the restricted natural join is $R'(\underline{a,b},c,b')$. The constraint is: for each tuple t in R', t[b] and t[b'] must agree. Note that the natural join of R_1 and R_2 is $R''(\underline{a,b},c)$ from R'' we cannot find the constraint.

Example 10. Let $\mathcal{R} = \{R_1(\underline{a,b,c,d},e,f), R_2(\underline{e},b,g), R_3(\underline{f,g}), R_4(\underline{g},h)\}$. The implicit constraints $P^b_{ab} + (ab \to e) \circ (e \to b)$ and $(ab \to e) \circ (e \to g) + (ab \to f) \circ (f \to g)$ can be transformed into

$$(R_1[a,b,e] \circ R_2[e,b'])[a,b,b'] \quad \text{(where b in } R_2 \text{ is renamed by b')}$$

and $$(R_1[a,b,e] \circ R_2[e,g])[a,b,g] + (R_1[a,b,f] \circ R_3)[a,b,g]$$

respectively, where "+" has the same meaning as in label expressions, which denotes a constraint.

Example 11 Let $\mathcal{R} = \{R_1(\underline{a},b,c,d,g), R_2(\underline{b,c},f), R_3(\underline{d},f)\}$. The label expression $((a \to b) \wedge (a \to c)) \circ (bc \to f) + (a \to d) \circ (d \to f)$ can be transformed into

$$(R_1[a,b,c] \circ R_2)[a,f] + (R_1[a,d] \circ R_3)[a,f].$$

If we let $R_1' = R_1[a,b,c,d]$ then it can be also transformed into

$$(R_1' \circ R_2)[a,f] + (R_1' \circ R_3)[a,f].$$

For simplicity, we can write it as

$$(R_1' \circ R_2 + R_1' \circ R_3)[a,f]$$

since there is no ambiguity to interpret it as the original expression.

Expressing an integrity constraint in terms of operations on relations allows us to establish the validity of an instance of the relations at any point in time. However, it is sometimes more desirable to check on every update that the resulting instance does not violate any integrity constraint.

Let R be an optimal set of relations with the functional dependencies $F(R)$. Let

$$\alpha = \alpha_1 + \alpha_2 + \ldots + \alpha_n \qquad (n \geq 2)$$

be a disjunctive normal form label of a functional dependency $A \rightarrow b \in F^+(R)$. Let $f_i : A \rightarrow b$ be the partial function defined by the term α_i at a time t, for $i = 1,2,\ldots,n$. Let $f_i' : A \rightarrow b$ be the new partial function defined by α_i after the data base instance is updated, for $i = 1,2,\ldots,n$. In order to check whether the updated data base instance satisfies the implicit constraint induced by α, we must merely ensure that for all $i = 1,2,\ldots,n$, if there exists an A-value a such that $f_i'(a)$ is defined and $f_i'(a) \neq f_i(a)$ then

(1) if there exists $j \in \{1,2,\ldots,n\}$, $i \neq j$, such that $f_j(a)$ is defined and $f_j'(a) = f_j(a)$, then $f_i'(a) = f_j(a)$,

(2) if for all $j \in \{1,2,\ldots,n\}$, $i \neq j$, $f_j(a)$ is undefined or $f'_j(a) \neq f_j(a)$, then $f'_i(a) = f'_\ell(a)$ whenever $f'_\ell(a)$ is defined and $\ell \in \{1,2,\ldots,n\}$.

This gives us a better way to check the implicit constraints because we need not check whether or not f'_i and f'_j agree for all $i \neq j$, $i, j = 1,2,\ldots,n$.

5. Conclusion

We have shown that an optimal set of relations may contain some implicit constraints. In fact, implicit constraints may even exist in optimal sets of relations in Boyce-Codd form, which were designed to reduce such dependencies between attributes [Ling 78b]. We have given an algebraic representation for describing the functional dependencies in the closure of a given set of functional dependencies. We have proved that we only need to find the implicit constraints induced by the functional dependencies of the form $K \rightarrow b$, where K is a primary key of some relation in the set and b is an attribute, and we can ignore the implicit constraints induced by the label expressions of the form $\alpha = \beta \circ \gamma$, where β and γ are also label expressions. An algorithm, which is based on a tool called functional dependency table, to find the minimum set of implicit constraints within a given optimal set of relations is given in [Ling 78b].

Acknowledgements

We are grateful to Professor Tom Maibaum for his helpful discussion and suggestions based on a previous version of this report.

Aho 77

Aho, A.V., Beeri, C., and Ullman, J.D. The theory of joins in relational data bases. 18th Annual Symposium on Foundation of Computer Science (1977).

Bernstein 75

Bernstein, P.A. Normalization and functional dependencies in the relational data base model. Ph.D. dissertation, Department of Computer Science, University of Toronto, (1975), CSRG-60.

Bourbaki 68

Bourbaki, N. Elements of Mathematics: Theory of Sets. Addison-Wesley Publ. Co., Reading, Mass., (1968).

Chamberlin 74

Chamberlin, D.D., and Boyce, R.F. SEQUEL: A Structure English Query Language. Proc. 1974, ACM SIGMOD Workshop on Data Description, Access and Control, 249-264.

Codd 70

Codd, E.F. A relational model for large shared data banks, CACM 13,6 (June 1970), 377-387.

Codd 71

Codd, E.F. Further normalization of the relational data base model. Courant Computer Science Symposium 6, Data Base Systems. R. Rustin (ed.) Prentice-Hall, (1971), 33-64.

Codd 74

Codd, E.F. Recent investigations in relational data base systems. IFIP 74, North-Holland, 1017-1021.

Date 77

Date, C. J. An Introduction to Database Systems. Second edition, Addison-Wesley Publishing Co., Reading, Mass., (1977).

Fadous 75

Fadous, R., and Forsyth, J. Finding candidate keys for relational data bases. ACM SIGMOD International Conference on Management of Data, (May 1975), 203-210.

Furtado 77

Furtado, A.L., and Sevcik, K.C. Permitting updates through views of data bases. Technical report, Pontificia Universidade Católica do Rio de Janeiro, Brasil (1977).

Hammer 75

Hammer, M.M., and McLeod, D.J. Semantic integrity in a relational data base system. Proc. International Conference on Very Large Data Bases, (1975), 25-47.

Kapali 75

Kapali, P.E., Eswaran, K.P., and Chamberlin, D.D. Functional specifications of a subsystem for data base integrity. Proc. International Conference on Very Large Data Bases, (1975), 48-68.

Ling 78a

Ling, T.W., and Tompa, F. Adequate definitions in third normal form. Dept. of Computer Science, University of Waterloo, Research Report CS-78-34, (July 1978).

Ling 78b

Ling, T.W. Improving data base integrity based on functional dependencies. Ph.D. dissertation, Dept. of Computer Science, University of Waterloo, (July 1978).

Osborn 78

Osborn, S.L. Normal forms for relational data bases. Ph.D. dissertation, Dept. of Computer Science, University of Waterloo, (1978), CS-78-06.

Stonebraker 75

Stonebraker, M. Implementation of integrity constraints and views by query modification. Proc. SIGMOD Conference (1975), 65-78.

Stanat 77

Stanat, D.F., and McAllister, D.F. Discrete Mathematics in Computer Science. Prentice-Hall Inc., Englewood, Cliffs N.J. (1977).