Integrating Data from Possibly Inconsistent Databases

Phan Minh Dung
Department of Computer Science
Asian Institute of Technology
PO Box 2754, Bangkok 10501, Thailand
dung@cs.ait.ac.th

Abstract

We address the problem of data inconsistencies while integrating data sets from multiple autonomous relational databases. We start by arguing that the semantics of integrating possibly inconsistent data is naturally captured by the maximal consistent subsets of the set of all information contained in the collected data. Based on this idea, we propose a simple and intuitive semantical framework, called the integrated relational calculus which is an extension of the classical relational calculus, for manipulating and querying possibly inconsistent data. We then show that our model generalizes the recently proposed model of flexible relational algebra of Agarwal, Keller, Wiederhold and Saraswat in the sense that the latter can be embedded into the former. We also shows that the flexible relational model is not capable to integrate correctly relations with more than one keys. We further argue that flexible relational model provides a rather weak query language. We then proves that for the databases with only one key the flexible model provides a correct integration of inconsistent data.

1. Introduction

A growing numbers of database applications need to jointly manipulate data from loosely coupled autonomous databases connected by high speed communication networks [12,10,5]. Due to their autonomy, the distribution of data in these databases tend to be arbitrary, often redundant and possibly inconsistent. This makes the development and maintenance of these applications costly and difficult since traditional models of databases like the relational model do not provide support for handling inconsistent data. The problem of dealing with inconsistent data is left to the application developer. But if the amount of data is huge this will become a formidable task.

Another option is to extend the classical data models to provide support for dealing with possibly inconsistent data. Recently Agarwal, Keller, Wiederhold and Saraswat [1] has forwarded an extension of the relational model, called the flexible relational model. But the flexible relational model suffers from a number of serious problems. The following example shows its inability to integrate possibly inconsistent relations if the associated relation schema has more than one key.

Example 1.1 *Let $\{employee, wife\}$ be a relation schema with two keys $\{employee\}$ and $\{wife\}$ with the former being the primary key. Let*

R_1

employee	wife
Terry	*Lisa*

R_2

employee	wife
Peter	*Lisa*

Integrating R_1, R_2 using flexible model results in a flexible relation:

P

employee	wife
Terry	$\{Lisa\}$
Peter	$\{Lisa\}$

Now asking the question "whose wife is Lisa ?"*, the flexible relational algebra will returns the incorrect answer $\{Terry, Peter\}$. In this example, there is an inconsistency among the data in R_1, R_2 due to the fact that $\{wife\}$ is a key. The flexible algebra fails to detect this inconsistency and hence provides the wrong answer. A correct answer must state that it is undetermined who is the husband of Lisa.*

The example shows that in general flexible algebra does not capture the intuitive semantics of integrating possibly inconsistent data sets from multiple autonomous databases. In this example, the intuitive semantics says that integrating R_1, R_2 will result in two possible scenarios represented by the following two relations:

R_1'

employee	wife
Terry	*Lisa*
Peter	$\perp$

R_2'

employee	wife
Terry	$\perp$
Peter	*Lisa*

where $\perp$ is the null value. The question "whose wife is Lisa ?" *is understood as asking: Give the name of the person who is the husband of Lisa in every possible scenario.*

Another problem of flexible relational model is that it provides a rather weak query language.

Example 1.2 *Consider the following two relations over the relation schema $\{employee, department\}$ with $\{employee\}$ as the primary key.*

R_1

employee	department
Terry	*CS*

R_2

employee	department
Terry	*Math*

Integrating R_1, R_2 using flexible model results in a flexible relation:

P

employee	department
Terry	$\{CS, Math\}$

For the question "who is employed in CS or Math ?" *represented by the selection formula*

$department = CS \vee department = Math$,

the expected answer is $\{Terry\}$. But flexible model will give $\emptyset$ as the answer meaning that it does not know who is working in CS or Math.

We may also ask "who is possibly employed in CS ?", *the expected answer is again Terry but there is no way to express this query using flexible algebra.*

In this paper, we restrict ourself on the problem of integrating data from multiple autonomous relational databases that may be mutually inconsistent. We assume that all other kinds of heterogeneities like ontologies, operating systems ect. have been resolved via a homogenizing veneer on each individual database. We start by arguing that the semantics of integrating possibly inconsistent data is naturally captured by the maximal consistent subsets of the set of all information contained in the collected data. Based on this semantics, we develop a query language called the integrated relational calculus that is a conservative extension of the classical relational calculus. We then study the relationship between flexible relational algebra and integrated relational calculus. We show the soundness of flexible algebra for the class of databases having exactly one key. We also show that for databases in this class, expressions in flexible algebra can be transformed into equivalent formula of integrated relational calculus. Due to the computational attractiveness of flexible algebra, this transformation could be viewed as a query optimization technique for a significant class of queries in integrated relational calculus. We will end with a discussion on open problems related to this work.

Dealing with incomplete and possibly inconsistent data is a much studied problem in the literature [7,9,3,11]. The main difference between our work and these works is that we study this problem in the presense of functional dependencies. In constrast, the works we know in the literature [7,9,3,11] on incomplete information in relational databases are based on the assumption that there are no integrity constraints between the data. Hence none of them can handle the problems discussed in example 1.1. The reason here is that these works view the semantics of a database with incomplete or possibly inconsistent data as a collection of complete and consistent databases containing no null values. This constrasts with our framework where the semantics of incomplete and possibly inconsistent databases is captured by the maximal consistent subsets of the set of all information contained in the collected data. These maximal consistent subsets are represented using Zaniolo's null value as no information [13].

2. Preliminaries: Null as No Information

Let S = (K,Z) be an arbitrary but fixed relation schema where K is the primary key and Z is the set of attributes not in K. We assume that S is in Boyce-Codd normal form where the set of keys of S is denoted by Key_S. Note that K always belongs to Key_S. The domain for each attribute $A \in K \cup Z$ is denoted by DOM(A). Note that the null value $\perp$ is not contained in DOM(A). Further, let $DOM(A)_\perp = DOM(A) \cup \{\perp\}$.

Definition 2.1 *(Tuples)*
A tuple over (K,Z) is a mapping assigning to each attribute $A \in K \cup Z$ an element in $DOM(A)_\perp$ where the value assigned to each attribute in the key K is not null.

We make an assumption that the values of the attributes in the primary keys are correct. Chatterjee et all [4] has studied the problem of dealing with inconsistency involving attributes in the primary key.

Definition 2.2 *(Conflicting tuples)*
Two tuples t,t' over S=(K,Z) are said to be conflicting *if there exists a key $K' \in Key_S$ such that for each $B \in K'$, $t(B) = t'(B) \neq \perp$ and there is $A \in K \cup Z$ such that $\perp \neq t(A) \neq t'(A) \neq \perp$.*

Similarly to Agarwal et all [1] we choose the value null to have the intepretation of no information [13]. Interpreting null as no information leads naturally to the following information-wise partial order $\sqsubseteq$ on $DOM(A)_\perp$:

For all $e, e' \in DOM(A)_\perp$: $e \sqsubseteq e'$ if and only if $e = \perp$ or $e = e'$.

For all tuples t,t' over (K,Z), we say that t is *less informative* than t', denoted by $t \sqsubseteq t'$, if and only if for each $A \in K \cup Z$, $t(A) \sqsubseteq t'(A)$.

Tuples $t_1, \ldots, t_n$ are said to be *joinable* if there exists an tuple t' such that each i, $1 \leq i \leq n$, $t_i \sqsubseteq t'$, i.e. t_i is less informative than t'. From the definition of tuple, it is clear that if t,t' are joinable then t[K] = t'[K]. If t and t' are joinable then *t + t'* (the sum of the information contained in t,t') is defined by $\forall A \in Z, (t + t')(A) = max\{t(A), t'(A)\}$.

Definition 2.3 *(Informative Closure)*

- *A set of tuples S over (K,Z) is* closed *if following conditions are satisfied:*
  - *For all $t, t' \in S$, if t,t' are joinable then t + t' also belongs to S.*
  - *For each $t \in S$, S contains each t' satisfying $t' \sqsubseteq t$*
- *The informative closure of S, denoted by $\hat{S}$ is the least closed relation containing S.*

Definition 2.4 *A set of tuples S is* consistent *if $\hat{S}$ contains no conflicting tuples*

Definition 2.5 *(Relations)*
A consistent set of tuples R over (K,Z) is a relation over (K,Z) if for all $p, p' \in R$, if $p[K] = p'[K]$ then p, p' coincide

The notion of being less informative is now extended to relations. A relation R is said to be *less informative* than (or subsumed by) a relation R' if for each tuple $t \in R$, there exist a tuple $t' \in R'$ such that $t \sqsubseteq t'$. Intuitively a relation is said to be less informative than another relation if each piece of information contained in the former is also contained in the later.

For each set of tuples S, the set of all maximal elements in $\hat{S}$ is denoted by $\overline{S}$. It is easy to see that for each relation R, $R = \overline{R}$.

For each set of tuples S, $\overline{S}$ is called the *relational representation* of S.

3. The Integrated Relational Model

Integrating data from multiple autonomous databases is understood as an operation for *collecting and processing* the information contained in this databases for the purpose of obtaining more information and in the case there is inconsistency, of being able to draw more reliable conclusions than those based on only one database.

The collecting step is easily done by taking the union of the relations. Let R,R' be relations over (K,Z). If the collected information from R and R' represented by $R \cup R'$ is consistent then the relation $R \cup R'$ represents the integration of information from R,R'. If $R \cup R'$ is inconsistent, a maximal consistent subset of the set of all information contained in $R \cup R'$ would be one possible admissible collection of information an user can get from integration. The semantics of the integration is then represented by the class of all possible admissible collections of information.

Now we formalize what we have just dicussed. The first task is to represent the possible admissible collections of information. A straightforward idea is to use a maximal consistent subsets of $R \cup R'$ to represent such collections. But the following example easily refutes this idea.

Example 3.1 *Consider the following relations*

R_1

employee	tel	salary
Terry	*5709*	*35*

R_2

employee	tel	salary
Terry	$\perp$	*20*

where $\{employee\}$ is the primary key and also the only key. One of the possible maximal consistent subsets of the set of all information contained in $R_1 \cup R_2$ is represented by the following relation

employee	tel	salary
Terry	*5709*	*20*

which is not a maximal consistent subset of $R_1 \cup R_2$.

The informative closure of a relation R contains as much information as R but also contains an explicit representation for each representable piece of information in R. Hence it is clear that each maximal consistent subset of the set of all information contained in $R \cup R'$ can be represented by a maximal consistent subset of the set $\hat{R} \cup \hat{R}'$.

Definition 3.2 *(Integration Semantics)*
Let $R_1, \ldots, R_n$ be relations over the relational schema (K,Z).

- *A possible integration of $R_1, \ldots, R_n$ is defined as the relational representation of a maximal consistent subset of $\hat{R}$ where $R = \hat{R_1} \cup \ldots \cup \hat{R_n}$.*
- *The collection of all possible integrations of $R_1, \ldots, R_n$ is defined as the semantics of integrating $R_1, \ldots, R_n$ denoted by $Integ(R_1, \ldots, R_n)$,*

Example 3.3 • *It is not difficult to see that in 1.1,* $Integ(R_1, R_2) = \{R_1', R_2'\}$

- *In example 3.1, it is not difficult to see that* $Integ(R_1, R_2)$ *consists of the following relatiosns*

employee	tel	salary
Terry	*5709*	*35*

employee	tel	salary
Terry	*5709*	*20*

4. Extending Relational Calculus for Querying Integrated Data

Example 4.1 *Consider the following relations*

R_1

employee	salary
Terry	*35*
Peter	*28*

R_2

employee	salary
Terry	*20*
Peter	*25*

Then $Integ(R_1, R_2) = \{R_1, R_2, W_1, W_2\}$ *with*

W_1

employee	salary
Terry	*35*
Peter	*25*

W_2

employee	salary
Terry	*20*
Peter	*28*

Each of the possible integrations of R_1, R_2 *can be viewed as containing information about a possible world. Now consider the following queries:*

Q_1: Give the names of all employees whose salary is possibly less than 30.

Q_2: Give the names of all employees whose salary is less than 30.

The salary of a person is possibly less than 30 if there is a possible world in which this person's salary is less than 30. A person's salary is less than 30 if her salary is less than 30 in all possible worlds.

Hence the expected answer for first query is $\{Peter, Terry\}$ *while the expected answer for the second query is* $\{Peter\}$.

Now we want to define the integrated relational calculus for formulating queries like Q_1, Q_2. The integrated relational calculus is an extension of the classical domain relational calculus with a modal operator $\mathcal{K}$ to allow us to "quantify" over the set of possible worlds.

Formally, The integrated relational calculus over a relation schema S = (K,Z)[1] is a first order modal language with a single modal operator $\mathcal{K}$ constructed in the usual way from the atomic formulas with S as a predicate symbol, a countably infinite set of variables and a set of constants where the null value $\perp$ is viewed as a constant. The atomic formulas are

- either a literal $S(X_1, \ldots, X_n)$ where $X_1, \ldots, X_n$ are variables or constants
- or an arithmetic comparision $X\theta Y$ where X,Y are variables or constants and θ is one of the arithmetic comparision operators $=, \neq, >, \geq, <, \leq$

Note that we often use attribute names as variables.

A *possible world* over a relational schema S is defined as a relation over S.

We define now the "truth" ($\models_t$) and "falsity" ($\models_f$) of formulas in the integrated relational calculus w.r.t. a possible world W and a set of possible worlds $\mathcal{W}$. *The information-wise intuition behind the "truth" of a formula F in the integrated relational calculus w.r.t* $(\mathcal{W}, W)$ *is that there is enough information in* $(\mathcal{W}, W)$ *to validate F. Similarly* $(\mathcal{W}, W) \models_f F$ *means that there is not enough information in* $(\mathcal{W}, W)$ *to validate F.*

Definition 4.2 *(*$\models_t, \models_f$*)*

- $(\mathcal{W}, W) \models_t S(\vec{a})$ *iff the tuple* $\vec{a}$ *belongs to W. Note that* $\vec{a}$ *may contain a null value.*
- $(\mathcal{W}, W) \models_t c\theta c'$ *iff* $c\theta c'$ *holds where* θ *is one of the six arithmetic operators, and c,c' are arithmetic constants.*
- $(\mathcal{W}, W) \models_t \mathcal{K}F$ *iff for each* $W' \in \mathcal{W}$, $(\mathcal{W}, W') \models_t F$
- $(\mathcal{W}, W) \models_t \exists x.F(x)$ *iff there exists a constant* $c \neq \perp$ *such that* $(\mathcal{W}, W) \models_t F(c)$
- $(\mathcal{W}, W) \models_t \neg F$ *iff* $(\mathcal{W}, W) \models_f F$
- $(\mathcal{W}, W) \models_t F \wedge F'$ *iff* $(\mathcal{W}, W) \models_t F$ *and* $(\mathcal{W}, W) \models_t F'$

- $(\mathcal{W}, W) \models_f S(\vec{a})$ *iff* $\vec{a} \notin W$
- $(\mathcal{W}, W) \models_f c\theta c'$ *iff* $c\theta c'$ *does not hold.*
- $(\mathcal{W}, W) \models_f \mathcal{K}F$ *iff for some* $W' \in \mathcal{W}$, $(\mathcal{W}, W') \models_f F$
- $(\mathcal{W}, W) \models_f \exists x.F(x)$ *iff for each constant c s.t.* $c \neq \perp$, $(\mathcal{W}, W) \models_f F(c)$.

[1] for simplicity, we define the integrated relational calculus over only one relation schema. But the definition can be easily extended for any database schema

- $(\mathcal{W}, W) \models_f \neg F$ *iff* $(\mathcal{W}, W) \models_t F$
- $(\mathcal{W}, W) \models_f F \wedge F'$ *iff* $(\mathcal{W}, W) \models_f F$ *or* $(\mathcal{W}, W) \models_f F'$

It is easy to see that the truth of formulas not containing $\mathcal{K}$ does not depend on $\mathcal{S}$ while the truth of formulas of the form $\mathcal{K}$F does not depend on W.

The following example demonstrates that the above definition captures the information-wise intuition of the relations $\models_t, \models_f$

Example 4.3 *Consider*

R

employee	tel
Terry	$\perp$

Since null means no information, $(Terry, \perp) \in R$ *means that there is no information whatsoever in R about whether or not Terry has a telephone. Consequently from the intuition of "...* $\models_f F$*" as "not enough information in ... to validate F", we expect that* $(\mathcal{W}, R) \models_f \exists x.S(Terry, x)$ *holds for every* $\mathcal{W}$*. Indeed, this is exactly what we get from the definition of* $\models_f$*.*

Definition 4.4 *(Query)*
A query denoted by a formula F($\vec{x}$*) is expressed by*

$$\{\vec{x} \mid F(\vec{x})\}$$

Now we can define the answer to a query.

Definition 4.5 *(Answers)*
Let Q be a query $\{\vec{x} \mid F(\vec{x})\}$*, and W be a world and* $\mathcal{W}$ *be a set of worlds.*

- *The answer to Q w.r.t.* $(\mathcal{W}, W)$ *denoted by* $Ans_Q(\mathcal{W}, W)$*, is defined as the set of all tuples* $\vec{c}$ *such that* $(\mathcal{W}, W) \models_t F(\vec{c})$*.*
- *The answer to Q w.r.t.* $\mathcal{W}$ *denoted by* $ANS_Q(\mathcal{W})$ *is defined by*

$$ANS_Q(\mathcal{W}) = \bigcup \{Ans_Q(\mathcal{W}, W) \mid W \in \mathcal{W}\}$$

- *For short, we often write* $ANS_Q(R_1, \ldots, R_n)$ *for* $ANS_Q(Integ(R_1, \ldots, R_n))$

Example 4.6 *The two queries* Q_1, Q_2 *in example 4.1 are denoted respectively by* F_1, F_2 *where*

$$F_1(x) \equiv \exists z.S(x, z) \wedge z < 30$$
$$F_2(x) \equiv \mathcal{K}(\exists z.S(x, z) \wedge z < 30)$$

It is easy to see that

$$Ans_{Q_1}(R_1) = \{Peter\}$$
$$Ans_{Q_1}(R_2) = \{Peter, Terry\}$$

$$Ans_{Q_1}(\mathcal{W}) = \{Peter, Terry\}$$

Since $Ans_{Q_2}(W) = \{Peter\}$ *for each each possible world W,* $Ans_{Q_2}(\mathcal{W}) = \{Peter\}$

5. Flexible Relational Algebra

In the previous chapter we have introduced the integrated relational calculus, which is an extension of the classical relational calculus, to provide a logical semantics and a query language for manipulating data from autonomous multiple databases. Agarwal,Keller,Wiederhold and Saraswat [1] pursuit another approach in which they propose the flexible relational algebra which is an extension of the classical relational algebra to deal with inconsistent data.

In the introduction we have given example showing that flexible relational algebra can give incorrect answer if there are more than one key. But flexible relational algebra is computationally attrative due to a compact and simple representation of the integrated data and a low-cost selection operation. This motivates us to find out reasonable suffcient conditions for the soundness of flexible algebra. We will show that for an important class of databases with exactly one key, flexible algebra is sound. We also will give a transformation to show that flexible relational algebra can be embedded into the integrated relational calculus.

Flexible relational algebra is based on the notion of cluple which is a cluster of compatible tuples. The semantics of a cluple is defined by a partial tuple obtained by merging the tuples in the cluple [1]. So for the sake of simplicity, we will identify in our recall of flexible algebra cluples with partial tuples.

Definition 5.1 *(Partial Tuple,Partial Relations)*

- *A partial tuple over a relational schema (K,Z) is a mapping from* $K \cup Z$ *which assigns to each attribute* $A \in K$ *exactly an element in DOM(A) and to each attribute* $B \in Z$ *either a nonempty finite subset of DOM(B) or the null value* $\perp$*.*
- *A set of partial tuples P over (K,Z) is said to be a partial relation over (K,Z) if for all* $p, p' \in P$*, if p[K] = p'[K] then p,p' coincide.*

Definition 5.2 *(Instances of Partial Tuple,Partial Relations)*

- *An instance of a partial tuple t is a tuple t' such that t[K] = t'[K] and for each* $A \in Z$*,*

$$t'[A] = \begin{cases} c \in t[A] & \text{if } t[A] \subseteq DOM(A) \\ \perp & \text{if } t[A] = \perp \end{cases}$$

- *An instance of a partial relation P is obtained by replacing each partial tuple in P by exactly one of its instances.*
- *The semantics of a partial relation P is defined by the set of its instances, denoted by Ins(P)*

Two partial tuples p,p' are said to be *compatible* if they contain data about the same entity, i.e. p[K] = p'[K]

Let $p_1, \ldots, p_n$ be compatible partial tuple over (K,Z). The merge of $p_1, \ldots, p_n$ denoted by $p_1 + \ldots + p_n$, is defined as a partial tuple p such that p[K] = $p_1[K]$ and for each $A \in Z$,

$$p(A) = \begin{cases} \perp & \text{if } p_i(A) = \perp \text{ for each i} \\ V & \text{otherwise} \end{cases}$$

where $V = \bigcup\{p_i(A) \mid p_i(A) \subseteq DOM(A)\}$.

Flexible model uses partial relations to represent the integration of possibly inconsistent relations. For example, the integration of the following relations

employee	*tel*
Terry	5709
Peter	5708

employee	*tel*
Terry	5700

with $K = \{employee\}$, is represented by the partial relation

employee	*tel*
Terry	$\{5709, 5700\}$
Peter	$\{5708\}$

The set of operations for flexible relational algebra defined in [1] includes union, selection, projection and cartesian product. In the following we will introduce union and selection. It is straitforward to extend the operations projection and Cartesian product of classical relational algebra to partial relations.

5.1. Union

The union of two partial relations in flexible algebra is obtained by merging the compatible partial tuples in them.

Definition 5.3 *Let P,P' be partial relations.*

$$P + P' = S_1 \cup S_2 \cup S_3$$

where $S_1 = \{p + p' \mid p \in P, p' \in P'$ *such that p, p' are compatible* $\}$, $S_2 = \{p \in P \mid$ *there exists no compatible tuples in P'* $\}$, $S_3 = \{p' \in P' \mid$ *there exists no compatible tuples in P* $\}$

In the flexible relational model, the integration of relations $R_1, \ldots, R_n$ *is defined as the union* $R_1 + \ldots + R_n$..

As example 1.1 shows, in general, $Ins(R_1 + \ldots + R_n) \neq Integ(R_1, \ldots, R_n)$. That means that in general, $R_1 + \ldots + R_n$ does not capture the intuitive semantics of integrating possibly inconsistent data from multiple databases. But if the primary key is the only key then $R_1 + \ldots + R_n$ is indeed a correct representation. The following theorem is one of the results of this paper.

Theorem 5.4 *Let S=(K,Z) with* $Key_S = \{K\}$. *Let* $R_1, \ldots, R_n$ *be relations over (K,Z). Then*

$$Integ(R_1, \ldots, R_n) = Ins(R_1 + \ldots + R_n)$$

Remark From now on until the end of this paper, we restrict ourself on relation schemas with exactly one key.

5.2. Selection

A selection formula over a set of attributes H is defined as a formula involving arithmetic operators $=, \neq, <, \leq, >, \geq$, the logical operators $\wedge, \vee$, and $\neg$, and operands that are constants or atributes from H. Note that the null value $\perp$ is viewed as a constant.

The truth (or satisfiability) ($\models_t$) and falsity ($\models_f$) of a selection formula F w.r.t. partial tuple p is defined as follows:

- $p \models_t A\theta A'$ iff $\forall c \in p(A), \forall c' \in p(A').\ c\theta c'$ holds
- $p \models_t A\theta c'$ iff $\forall c \in p(A).\ c\theta c'$ holds
- $p \models_t \neg F$ iff $p \models_f F$
- $p \models_t F \wedge F'$ iff $p \models_t F$ and $p \models_t F'$
- $p \models_t F \vee F'$ iff $p \models_t F$ or $p \models_t F'$

- $p \models_f A\theta A'$ iff $\neg\exists c \in p(A), \neg\exists c' \in p(A').\ c\theta c'$ holds
- $p \models_f A\theta c'$ iff $\neg\exists c \in p(A).\ c\theta c'$ holds
- $p \models_f \neg F$ iff $p \models_t F$
- $p \models_f F \wedge F'$ iff $p \models_f F$ or $p \models_f F'$
- $p \models_f F \vee F'$ iff $p \models_f F$ and $p \models_f F'$

Definition 5.5 *(Answers)*
Let F be a selection formula and P be a partial relation. The $\sigma_F(P)$ *is defined as the set of those partial tuples in P satisfying F.*

5.3. Transforming Flexible Relational Model into Integrated Relational Model

A selection formula F is said to be in conjunctive normal form (CNF) iff it is of the form $F_1 \wedge \ldots \wedge F_n$ such that no F_i contains $\wedge$ and negation applies only to individual comparision.

Now we want to give a transformation from flexible selection formula F into equivalent query Q_F in integrated relational calculus.

Definition 5.6 *Let S=(K,Z) be a relational schema and F be a seclection formula over $K \cup Z$ in CNF. Then*

$$Q_F = \{K, Z \mid S(K,Z) \wedge \mathcal{T}(F)\}^2$$

where $\mathcal{T}(F)$ is defined as follows

- $\mathcal{T}(L) = \mathcal{K}(\exists Z.S(K,Z) \wedge L)$ [3] *where L is an individual comparision or the negation of an individual comparision.*
- $\mathcal{T}(F \vee F') = \mathcal{T}(F) \vee \mathcal{T}(F')$
- $\mathcal{T}(F \wedge F') = \mathcal{T}(F) \wedge \mathcal{T}(F')$

We can give now one of the main results of this paper.

Theorem 5.7 *Let $R_1, \ldots, R_n$ be relations over a relational schema S = (K,Z) with $Key_S = \{K\}$ and P = $R_1 + \ldots + R_n$. Further let F be an arbitrary selection formula in CNF over $K \cup Z$. Then*

$$\sigma_F(P) = ANS_{Q_F}(R_1, \ldots, R_n)$$

It is clear that flexible relational algebra is fairly weak. It for example does not allow us to ask question like the first one in example 4.1. In general, the question as whether or not it is possible to extend the flexible relational algebra to capture the power of safe integrated relational calculus is left open.

6. Conclusions and Future Works

We have provided a simple and intuitive semantical framework for manipulating possibly inconsistent data from multiple autonomous databases. We then proposed the integrated relational calculus, an extension of the traditional relational calculus, as a query language. These results establish a semantical foundation for integrating and querying possibly inconsistent data. Based on this foundation, we showed that though the flexible relational algebra is not sound in general, it is sound for the important class of databases whose only dependencies are those determined by the primary key. Further we also showed that flexible relational algebra can be embedded into integrated relational calculus.

Reasoning with incomplete and inconsistent information has also been studied extensively in AI. The integrated relational model seems to be related to the frameworks proposed in [2,6,8] though we are not clear whether our $\mathcal{K}$ operator is more related to the "know"-operator in [8] or to the strong introspection operator in [6]. Further, null as no information is a kind of metadata about the database. And integrated relational model provides a simple framework for dealing with this sort of metadata. We are not aware of systems in AI which deals with "no information" metadata.

There are a number of problems which have been left open in this paper. The first one is to find an effective algorithm for query evaluation. For those queries which are equivalent to expressions in flexible relational algebra, techniques developed in flexible relational model can be applied. But since flexible relational model is rather weak, we probably have to look elsewhere for such algorithm. Another problem is to extend this integrated relational model for other kinds of null values.

Acknowledgements

We would like to thank four anonymous referees for their constructive criticisms.

This research was supported in part by EEC Keep in Touch Activity KIT011.

References

[1] S. Agarwal, A. M. Keller, G. Wiederhold, and K. Saraswat. Flexible relations: An approach for integrating data from multiple possibly inconsistent databases. *Proc. of ICDE'95*.

[2] C. Baral, S. Kraus, J. Minker, and V. S. Subrahmanian. Combining knowledge bases consisting of first order theories. *Proc. of 6th International Symposium on Methodologies for Intelligent Systems*.

[3] K. S. Candan, J. Grant, and V. S. Subrahmanian. A unified treament of null values using constraints. *Technical report, Uni. of Maryland*.

[4] A. Chatterjee and A. Segev. Rule based joins in heterogeneous databases. *Decision Support Systems, Vol 13, 1995*.

[5] P. Drew, R. King, D. McLeod, M. Rusinkiewicz, and A. Silberschatz. Report on the third workshop on semantic heterogeneity and interoperation in multidatabase systems.

[6] M. Gelfond. Strong introspection. *Proc. AAAI-91*.

[7] T. Imielinski and W. Lipski. Incomplete information in relational databases. *JACM, Vol 31, No 4, 1984*.

[8] V. Lifschitz. Nonmonotonic databases and epistemic queries. *Proc. of IJCAI'91*.

[2] Here "K,Z" denotes a list of all elements in K,Z and S(K,Z) is a atomic formula with predicate symbol S and variables from K,Z.

[3] "$\exists Z$" stands for an existantial quantification over each attribute (considered as variable) in Z

[9] W. Lipski. On semsntic issues connected with incomplete information databases. *ACM TODS, Vol 4, No 3, 1979*.

[10] W. Litwin, L. Mark, and N. Roussopoulos. Interoperability of multiple autonomous databases. *ACM Computing Surveys, Vol 22, No 3, Sep 1990*.

[11] P. Scheuermann and E. I. Chong. Role-based query processing in multidatabases systems. *EDBT'94, 95-108*.

[12] A. Sheth and J. Larson. Federated database systems for managing distributed heterogeneous and autonomous databases. *ACM Computing Surveys, Vol 22, No 3, Sep 1990*.

[13] C. Zaniolo. Database relations with null values. *Journal of Computer and System Sciences, 28,pp 142-166,1984*.