**Data Integration Inconsistencies Resolution**

Štuller, Július
2001

**Institute of Computer Science**
**Academy of Sciences of the Czech Republic**

# Data Integration Inconsistencies Resolution

Július Štuller

Technical report No. 847

October 2001

Pod Vodárenskou věží 2, 182 07 Prague 8, phone: (+420) 2 6605 3200, fax: (+4202) 858 57 89,
e-mail:stuller@cs.cas.cz

**Institute of Computer Science**
**Academy of Sciences of the Czech Republic**

# Data Integration Inconsistencies Resolution [1]

Július Štuller

Technical report No. 847

October 2001

Abstract:

The process of knowledge discovery through data mining usually encounters at certain stage a need to integrate various data (bases). The general problem of data (bases) integration has been studied for more than two decades and many interesting results have been obtained. The emphasis was usually on schema integration, (logical) integrated views approaches or querying integrated databases, and relatively few work has been oriented towards data integration itself where the problems can be even more complicated.
Such a complication can be caused, for instance, by a possible occurrence of inconsistencies appearing in a data (bases) integration process. Inconsistencies in data can naturally invalidate them and so the data mining process on such erroneous data can lead to inaccurate or even wrong results, and consequently also to inadequate, wrong and sometimes bad or very costly decision(s).
And this is one of the main reasons why we would like to eliminate these inconsistencies as much as possible, or - in other words - to minimize their occurrence. Although a universal tool solving such a task is not yet available, and it is very questionable whether it will be ever available, probably nobody would doubt about the usefulness of a tool partially solving the task.
Moreover, the problem can easily become even more complicated as, for instance in the case of Internet data integration which may be considerably different from the classical – theoretical – database integration, the very usual case is the one in which we have to integrate data without having their complete description, where the data semantics are not given explicitly, or even data that may be incomplete, imprecise, etc. In such a case the methods and results of the database theory on database integration obtained in the last 20 years may be rather difficult to apply. On the other hand, some of the newly emerging areas – like, for instance, the soft computing with its approach sacrificing rigorousness and optimality versus fuzziness and suboptimality – may be promising.

Keywords:
Knowledge Discovery, Data Mining, Data Warehousing, Data Integration, Data Bases Integration, Integrity Constraint, Inconsistency

[1]This work was supported by the grant GA 201/00/1489 of the Grant Agency of the Czech Republic: *SOFT COMPUTING – Theoretical Foundations and Experiments.*

# Contents

# 1 Introduction

The data mining tools often require a specific format of data over which we want to do the data mining process. For instance the GUHA+ - package we use in our institute is a Windows implementation of the General Unary Hypothesis Automaton (GUHA) method [1, Hájek & Havránek (1978)] which itself has its origins in the late sixties [2, Hájek et al. (1966)] in exploratory data analysis and which works, in principle, over a data matrix into which the data have to be transformed. The easiest way to fill in this matrix is to import, for instance for the Microsoft family of products, either single Excel or Access table (in actual Office 2000 Suite both limited to 255 columns which turned out to be a constraint in our applications). And here we immediately face the following general problem: At a certain stage of knowledge discovery through data mining, at last in the process of the data mining itself, a need for (partial) data (bases) integration usually appears – although one can certainly do the data mining on a single database, or even on a single (relational) table, the integration of its building blocks (for instance relations in a relational database) will be very probably required.

The general problem of integration of data (bases) has been studied for almost two decades (see next section for more details). We will see later that in the case of the knowledge discovery from the Internet through data mining it can be considerably different from the theoretical premises developed in the last twenty years. And the possible inconsistencies occurring in such cases during the data integration can have important consequences: the wrong decisions taken – based on erroneous results of the data mining process, caused by the invalidated data due to their inconsistencies – can be really very expensive.

And this is exactly one of the main reasons why the possible occurrence of any inconsistencies appearing in a data integration process is something we would like to eliminate as much as possible, or – in other words – to minimize their occurrence. Although a universal tool solving such a task is not yet available, and it is very questionable whether it will be ever available, probably nobody would doubt about the usefulness of a tool partially solving the task.

Moreover, the problem can easily become even more complicated as, specially in the case of the Internet data integration which may be considerably different from the classical – theoretical – data integration, the very usual case is the one in which we have to *integrate data without having their complete description*, where the *data semantics are not given explicitly*, or even data that may be *incomplete, imprecise*, etc. And so in such cases the methods and the results of the general database theory on data (bases) integration obtained in the last twenty years may be rather difficult to apply.

On the other hand, some of the recently emerging new Computer Science paradigms – like, for instance, the **Soft Computing** with its *approach sacrificing* the classical mathematical fundamentals as the **rigorousness** and the **optimality** *versus* the **fuzziness** and the **suboptimality** – may be promising in the future.

The report is structured as follows: in the next section we will give a brief overview of the related work, then in section 3 we will specify the notion of the inconsistency (in a database), in section 4 we will formulate the problem, in section 5 we will precise what are the basic integration operations, in section 6 we will present a classification of the inconsistencies, in section 7 we will formulate the existence conditions for inconsistencies, in section 8 we will describe a procedure for inconsistencies resolution and in section 9 we will draw some conclusions.

## 2 Related work

The general process of the integration of data (bases) has been studied from the beginning of eighties. In most cases the *emphasis* was on:

- The **schema integration** - see for instance [3, Batini & Lenzerini (1984)] or a nice survey of the first "manual" methods in [4, Batini et al. (1986)] as ones of the first papers, and [5, Garcia-Solaco et al. (1995)], [6, Ramesh & Ram(1997)], [7, Santucci (1998)], [8, Tseng et al. (1998)], [9, Poulovassilis & Mc.Brien (1998)], [10, Kwan & Fong (1999)] or [11, Palopoli et al. (2000)] as ones of the more recent.
  While in the mid eighties *most of the researchers have suggested performing the integration activity as a part of the conceptual design step* (see [4] where the authors perceived the schema integration mainly as an integral part of database design methodologies, and pointed out that *a majority of the* – schema integration – *methodologies fall into the class of* **view integration** *methodologies*) some fifteen years later we can see a slight shift – according to [11]: the *scheme integration is devoted to producing a global conceptual scheme from a set of heterogeneous input schemes*.
- *(Logical) Integrated* **views** approaches. For instance [12, Gupta et al. (1986)] introduced the notion of *self-maintainable* **views** that can be maintained using only the contents of the view and the database modifications, without accessing any of the underlying databases.
  Their **integrated view** is materialized and stored in a database and queries on the view are then answered directly from the stored view.
  Relatively recent paper [13, Ullmann (2000)] reviews the formal basis of logical views techniques and shows they are closely related to containment algorithms for conjunctive queries.
- **Querying** integrated databases. For instance in [14, Arens et al. (1993)] the authors showed how a query at the domain level can be mapped into a set of queries to individual information sources (by generating and executing a plan for accessing the appropriate information sources) and presented algorithms for automatically improving the efficiency of queries using knowledge about both the domain and the information sources.
  More recent work [15, Arenas et al. (1999)] considered the problem of the logical characterization of the notion of the consistent answer in a relational database that may violate given integrity constraints (typicaly e. g. in a data warehouse containing data coming from different sources with some of them not satisfying the given integrity constraints).

From this point of view we have to admit that *much less attention* was given to the **integration of data** *themselves* (see [16, Hull & Zhou (1996)] or [17, Orlowska et al. (1997)] ). In [16], for instance, the **data integration** is based on the so called "Squirrel mediators" which can support besides the traditional *virtual* and *materialized view* approaches also hybrids of them.
The **consistency** of the *integration environment* is in [16] defined in terms of the *validity*, the *chronology* and the *order preservation*.

Generally, in the context of databases systems, the **consistency** is studied within the database system recovery, more precisely when dealing with database states where we want to keep database in a consistent state. According to [18, Date (2000), page 454, footnote]: *consistent means "satisfying all known integrity constraints"*. In [15] the consistency is defined via *consistent (relational) database instance* as the one "satisfying the (given) set of integrity constraints".

The **inconsistencies** in the process of database integration are similarly in most cases considered, if they are considered at all, only from the point of view of the schema integration (e. g. [7, Santucci (1998)]), usually for the *functional* or the *relational data models*. Generally they are discussed in the frame of various conflicts (structural or not: see [19, Lee et al. (1995)]; an other conflict taxonomy can be found in [9, Poulovassilis & Mc.Brien (1998)]). Their classification is, if done at all, very elementary (in [7], for instance, simply "inconsistent data") .
In [20, Hunter (2000)] the author developed more formal theory of inconsistency in the context of structured text (databases).
The paper [21, Castro et al. (1998)] presents an approach based on a probabilistic measure to be used in detection of inconsistencies in knowledge base systems.

In the field of Knowledge discovery from Databases an interesting description of an integrated environment acting as a software agent for discovering correlative attributes of data objects from multiple heterogeneous resource was given recently in [22, Chen et al. (2001)].

## 3 Inconsistency

Respecting the (classical mathematical) logic (see e.g. [23, Wang (1962)], page 27) definition:
*A system is said to be consistent (or free from contradiction) if there is no sentence p of the system such that both p and not-p are theorems*,
let us start by giving a definition of an inconsistency in a database:

**Definition 1** A database has an **inconsistency** if the data it contains yield under the given interpretation at least one contradiction.

The interpretation of the data in a database is given by their **semantics** which are, usually – at least partly, stored as *meta-data* [24, Etzion & Dahav (1998)] in the same database system.
From a mathematical logic point of view, these meta-data present an (axiomatic) theory $T$ expressing the so-called "background knowledge".
Consequently, our definition means a database has an inconsistency if the data it contains are inconsistent with the theory $T$, or – in other words – the union of the theory $T$ and of the data contains a contradiction.

The meta-data can be also viewed as the knowledge about the reality we are trying to capture in a database system and accordingly we can speak about a specific knowledge base which can be again studied for inconsistencies occurrence. In this report we will not go in more depth on this theme now but we will come back to it in the Conclusion section noting that similar procedures can be applied to the meta-data as to the primary data in a database system.

**Example 1**

| **Name** | **Year** |
|---|---|
| Jaromir Jagr | 1972 |
| Jaromir Jagr | 2001 |
| Mario Lemieux | 1965 |

In the first interpretation let the **Year** represents the *year of the birth* of a person with the corresponding **Name** (and let us further suppose the **Name** "Jaromir Jagr" represents a unique person). In the second interpretation let the **Year** represents the corresponding *important year(s) in the life* of a person with the corresponding **Name**.
While *without any interpretation we cannot decide at all whether there is or not a contradiction in our database*, under the *first interpretation* the given data *yield naturally a contradiction* (no person can be born in two different years, and, as a *consequence, in this concrete case, at least one datum* — year 1972 or 2001 — *must be incorrect*) and the *second interpretation* yields apparently *no contradiction*.
Summarizing, it is the *interpretation which decides whether the data yield or not inconsistency* and this is the reason why in the definition of a database inconsistency the given interpretation of data plays the *principal role*.

*Remark 1* In [18, Date (2000), page 454, footnote] the author wrote:
*... consistent does not necessarily mean correct: a correct state must necessarily be consistent, but a consistent state might still be incorrect, in the sense that it does not accurately reflect the true state of affairs in the real world.*
The author summarize by stating:
*"Consistent" might be defined as "correct as far as the system is concerned."*

We shall see later that, in general, the inconsistency says very little about the *correctness* of data.

**Definition 2** The concrete data of a given database which yield a contradiction will be called **inconsistent data**.

**Notation 1** Let $\mathcal{B}$ be a database, $\Delta$ the given interpretation of data in $\mathcal{B}$.
We will denote by $\mathcal{I}^{\Delta}(\mathcal{B})$ the inconsistent data of $\mathcal{B}$, or simply – in case of no possible ambiguity – $\mathcal{I}(\mathcal{B})$.

**Example 2** Under our first interpretation the inconsistent data are:

| Name | Year |
|---|---|
| Jaromir Jagr | 1972 |
| Jaromir Jagr | 2001 |

# 4 Formulation of the Problem

We will start by studying the conditions for the existence of inconsistencies during the process of the integration of several databases under the following natural logical assumption:

**A1:** *Each of the databases to be integrated has no inconsistent data.*

Furthermore, without any lost of generality, but for reasons of simplification, for the possibility to formalize the problem in an elegant manner, having in mind the current situation in the area of the database technologies where the *relational data model* prevails, and knowing that the others models can be, at least in theory, transformed into it, we will suppose that:

**A2:** *All the databases to be integrated are relational ones:*
Let $\mathcal{B}_i$, $i \in \widehat{m}$, be $m$ *relational databases* to be integrated ( $m \geq 2$ ), each consisting of $k_i$ *relations* $R^i{}_j = \langle A^i{}_j, D^i{}_j, T^i{}_j \rangle$.
(See Appendix for notations and definitions.)

We want to *design a methodology* and *propose a procedure* both of which will aim at the *elimination* of the *inconsistencies* when trying to integrate some of the databases $\mathcal{B}_i$.

# 5 Integration operations

From all the usual basic relational operations (and operators) the only ones which can contribute to the process of the integration of databases, and so could lead to possible inconsistencies, are the "update" operations, namely:

- the *unions* of the relations
- the *joins* (and the corresponding *compositions*).

**Definition 3** The following relational operations: *unions*, *(equi - ) joins* and *(equi - ) compositions* will be called the **integration operations**.

**Notation 2** We will use the symbol $\int$ to denote any integration operation without specifying exactly if it is an union, a join or a composition.

**Notation 3** We will use the notation $\int_{i=1}^{m} \mathcal{B}_i$ to denote the integration of databases $\mathcal{B}_i$ without specifying explicitly what integration operation(s) were/are/will be used on the appropriate relations $R^i{}_j$.

## 5.1 Union of the Relations

In order to be able to make the union of the relations $R^{i_j}{}_{q_j}$ we must first suppose they all have the same cardinality, say $k$ :

**A3:** $(\exists k \geq 1)\,(\exists s \geq 2)\;\forall j \in \widehat{s})\,(\exists \mathcal{B}_{i_j})\,(\exists R^{i_j}{}_{q_j} \in \mathcal{B}_{i_j})\,(\mid A^{i_j}{}_{q_j} \mid = k)$

*Remark 2* We can always find, by successive projections, the corresponding (sub) relations (of some $R^{i_j}{}_{q_j}$) with the required property.

Furthermore, for simplification, we will suppose the relations ${R^{i_j}}_{q_j}$ are defined over the same relational schema $\mathcal{S}$ :

**A4:** $$(\forall j \in \widehat{s})\,(\,{R^{i_j}}_{q_j} \sqsubset \mathcal{S} = \langle\, A\,,\, D\,\rangle\,)$$

**Example 3**

| $R_1$ | |
|---|---|
| **Name** | **Position** |
| Jordan | player |

| $R_2$ | |
|---|---|
| **Name** | **Position** |
| Jordan | owner |

| $R = R_1 \cup R_2$ | |
|---|---|
| **Name** | **Position** |
| Jordan | player |
| Jordan | owner |

Even in this very simple example, given the interpretation by a sport context of the basketball, without further supplementary information about this interpretation, it is impossible to decide whether an inconsistency appeared in the process of the integration of databases. Such a supplementary information is expressed in database systems as one or several *integrity constraint(s)* (see e.g. [25, Pernul et al. (1998)], [26, de Brock (2000)], [27, Zviran & Glezer (2000)]). We will suppose that we have such an integrity constraint. Let it be the only one:

*Every value of the attribute* **Name** *is associated with no more than one value of the attribute* **Position**. (A particular case of a functional dependency $R$ : **Name** $\rightarrow$ **Position** )

**Definition 4** We will call the data of the database $\mathcal{B}$ not satisfying the given set of the integrity constraints $\Sigma$ the **inconsistent data** *with respect to the set of the integrity constraints* $\Sigma$ and denote it by $\mathcal{I}_\Sigma(\mathcal{B})$ .

*Remark 3* In general the following inclusion holds: $\mathcal{I}_\Sigma(\mathcal{B}) \subset \mathcal{I}^\Delta(\mathcal{B})$ .

The situation in the **Example 3** can be formally rewritten in the following way:

$$((\forall i \in \widehat{2})\,((\mathcal{B}_i = \{\,R_i\,\}) \wedge (\mathcal{I}_\Sigma(\mathcal{B}_i)) = \emptyset))) \wedge (\textstyle\int_{i=1}^{2} \mathcal{B}_i = R = R_1 \cup R_2\,)$$
$$\wedge\,(\mathcal{I}_\Sigma(\textstyle\int_{i=1}^{2} \mathcal{B}_i) = R)$$

**More** we are able to *describe precisely the semantics of data* (and by this also *their interpretation*) in the form of the *appropriate integrity constraints* (and our *database system* **should** *be able to process* **all** *of them*), **more** we can *expect to automatize the process of discovering the inconsistencies* in the integration of databases.
The **ideal** *situation is the one in which we can consider the given set of integrity constraints as completely describing the semantics of data* (as for instance in [15] which allowed the authors to reduce the notion of the inconsistency to the following definition: *A database instance $r$ is consistent if $r$ satisfies $IC$* – the given set of integrity constraints – *in the standard model-theoretic sense, that is $r \vDash IC$ ; $r$ is inconsistent otherwise*).
In such a (ideal) case the following equality holds: $\mathcal{I}^\Delta(\mathcal{B}) = \mathcal{I}_\Sigma(\mathcal{B})$

The *contrary naturally leads* to a greater extent of *manual procedures*.
In the recent years there have been proposed some heuristics for searching of inconsistencies (see e.g. [21, Castro & Zurita (1998)]).

Returning again to our **Example 3** , we have seen that the inconsistent data (with respect to the given set of the integrity constraints) of the integrated database are equal to the *whole integrated database*.
Our final **goal** is to *minimize the inconsistencies* in the integrated database or, in other words, to *minimize the inconsistent data*. Naturally, the appropriate integrity constraints can largely help us in this and so we will always start by minimizing the inconsistent data *with respect to the given set of the integrity constraints*.
Unfortunately – specially in the case of the Internet data – the situation may be more complicated as the required helpful integrity constraints are very often missing or incomplete.

Now we will introduce some notions and notations:
Let us denote by $\Im$ the set of all the possible *integrity constraints* over the given *universe of discourse* $\mathcal{U} = \mathcal{D}(A)$

where: $A = \bigcup_{i=1}^{m} \bigcup_{j=1}^{k_i} A^i{}_j$ and $\mathcal{U} = A \cup (\bigcup_{i=1}^{m} \bigcup_{j=1}^{k_i} D^i{}_j(A^i{}_j))$

and by $I$ a subset of the set $\Im$.
Further denote by $\mathcal{R}(I)$ the set of all the relations over given universe of discourse satisfying $I$.
Generalizing the **Example 3** it is obvious that the following **Lemma 1** holds:

$$((\exists s \geq 2)(\forall j \in \widehat{s})(\exists \mathcal{B}_{i_j})(\exists R^{i_j}{}_{q_j} \in (\mathcal{B}_{i_j} \cap \mathcal{R}(I)))) \not\Rightarrow (\bigcup_{j=1}^{s} R^{i_j}{}_{q_j} \in \mathcal{R}(I))$$

*Remark 4* The *union*, to be **meaningful**, should be done only after a thorough *semantical justification* and *verification*, because the *syntactical equality* of the *attributes*, and of the *corresponding domains*, may be **misleading**, especially in the case of such *overloaded concepts* like *name*, *number*, *year*, etc.

Now we are ready to present a **methodology** of *resolving the inconsistencies in the process of the integration* – for the moment only by unions – *of databases*.
This methodology consists of *four steps* which we will describe and further refine in the following subsection.

## 5.2 The IFAR Methodology

STEP 1: **I***ntegrate* the databases $\mathcal{B}_k$ : $\int_{k=1}^{m} \mathcal{B}_k$

STEP 2: **F***ind* the set of inconsistent data: $\mathcal{I}(\int_{k=1}^{m} \mathcal{B}_k)$

STEP 3: **A***nalyze* the set $\mathcal{I}(\int_{k=1}^{m} \mathcal{B}_k)$ in order to find:

- *Inconsistent data with respect to the given set of the integrity constraints* $\Sigma$: $\mathcal{I}_\Sigma(\int_{k=1}^{m} \mathcal{B}_k)$
  $(\exists i \in \widehat{m})(\exists j \in \widehat{k_i})(\exists R^i{}_j = \langle A^i{}_j, D^i{}_j, T^i{}_j \rangle)(\exists t \in T^i{}_j)(t \not\models \Sigma)$
  Such a $t$ may not represent correctly a fact from the reality we are trying to capture in a database – in the relation $R^i{}_j$ (In our Example 3 it could mean that either Jordan is not a *player* or that he is not an *owner*.).
- *Wrong integrity constraints*:
  Some of $\mathcal{I}_\Sigma(\int_{k=1}^{m} \mathcal{B}_k)$ being correct could imply some integrity constraints from $\Sigma$ may be wrong – they may not correctly reflect the reality we are trying to model (In our Example 3 it could mean that there may be *more than one Position associated with one Name*.)
- *Wrong descriptions* of data:
  Some of $\mathcal{I}_\Sigma(\int_{k=1}^{m} \mathcal{B}_k)$ being correct could imply some attributes (description) are wrong (In our Example 3 it could mean, for instance, that datum "owner" is not a – value of the attribute – *Position*, but it should be a – value from an another attribute – *Function*.).

STEP 4: **R***esolution* of the inconsistencies:

- The result of the *"correction of data"* should be *new relations* $\widetilde{R^i{}_j}$ (without *incorrect – wrong – data*) over which we will do *integration* $\int_{i,j} \widetilde{R^i{}_j}$.
  The incorrect data should be discovered and corrected at the *data integration* stage.
- The result of the *"correction of integrity constraints"* should be a *new set of integrity constraints* $\widetilde{\Sigma}$ (without *wrong integrity constraints*).
  (At least some of) the wrong constraints should be discovered and their correction should be performed at the *schema integration* stage.
- The *"correction of attributes"* – usually – means the *renaming* of the *wrong attributes*.
  It should be done only after a thorough (*semantical*) analysis of data corresponding to the incorrect attributes.
  These incorrect attributes should be discovered and their renaming should be performed best again at the *schema integration* stage.

## 5.3 $\Pi$ - Unions

Next we will suppose that the relations $R^{i_j}{}_{q_j}$ are defined over such different relational schemata $\mathcal{S}^{i_j}{}_{q_j} = \langle A^{i_j}{}_{q_j}, D^{i_j}{}_{q_j} \rangle$ that there exist appropriate permutations $\pi^{i_j}{}_{q_j}$ in $|\widehat{A^{i_j}{}_{q_j}}|$ that the following holds:

**A5:** $\bigcap_{j=1}^{s} D^{i_j}{}_{q_j} ( \pi^{i_j}{}_{q_j} ( A^{i_j}{}_{q_j} ) ) \neq \emptyset$

**Example 4**

| $R_1$ | |
|---|---|
| **Name** | **Position** |
| Lemieux | player |

| $R_2$ | |
|---|---|
| **Name** | **Function** |
| Lemieux | owner |

| $R = R_1 \cup_\pi R_2$ | |
|---|---|
| **Name** | **Post** |
| Lemieux | player |
| Lemieux | owner |

The prerequisite is the existence of the permutations $\pi^{i_j}{}_{q_j}$ in $|\widehat{A^{i_j}{}_{q_j}}|$ which must be semantically justifiable for the concrete databases – relations: in our example we presuppose that the (names of the) attributes **Position** and **Function** are synonyms (i.e. they are semantically equivalent).

Relaxing the condition **A4** (about the relations one wants to make an union over being defined over the same relational schema) into weaker condition **A5** requiring the existence of permutations $\pi^{i_j}{}_{q_j}$ such that there exists the $\pi$ - union of relations $R^{i_j}{}_{q_j}$, one can obtain by similar reasoning we used to the union of relations the same sources of possible inconsistencies:

- *Inconsistent data with respect to the given set of the integrity constraints*
- *Wrong integrity constraints*
- *Wrong descriptions* of data.

and so the **IFAR** *methodology* can be used again.

The analogy of the **Lemma 1** for the $\pi$ - unions is the

**Lemma 2**: $( (\exists s \geq 2) (\forall j \in \widehat{s}) (\exists \mathcal{B}_{i_j}) (\exists R^{i_j}{}_{q_j} \in (\mathcal{B}_{i_j} \cap \mathcal{R}(I)) ) ) \not\Rightarrow ( \bigcup_{\pi\, j=1}^{s} R^{i_j}{}_{q_j} \in \mathcal{R}(I) )$

*Remark 5* In case of a $\pi$ - union, to obtain **meaningful** *results*, one should be *even more careful* to *semantically* well *justify* the real meaning of performing the operation of a $\pi$ - union.

*Remark 6* Although for *n-ary* relations we can have theoretically up to $n!$ possibilities (in other words: $n!$ different $\pi$ - unions) to integrate them using the $\pi$ - unions, in practice this number is considerably smaller due to:

- the *incompatibility* between certain attributes.
- the *fact that semantically meaningful is usually* **only one** *of them.*

## 5.4 ( Equi - ) Joins

In the following we will study the properties of the joins in the process of the integration of the (relational) databases.
We will begin by *illustrating* the **difference** between the *integration* by performing:

- one of the *joins* (the natural one) of the relations and
- one of the *unions* (the $\pi$ - *union*) of the same relations.

**Example 5**

| $R_1$ | |
|---|---|
| **Mother** | **Son** |
| Eve | John |

| $R_2$ | |
|---|---|
| **Mother** | **Daughter** |
| Eve | Anne |

| $R = R_1 * R_2$ | | |
|---|---|---|
| **Mother** | **Son** | **Daughter** |
| Eve | John | Anne |

| $R = R_1 \cup_\pi R_2$ | |
|---|---|
| **Mother** | **Child** |
| Eve | John |
| Eve | Anne |

Depending on the every concrete situation one must choose the *best appropriate operation* to perform the integration of the databases.
For instance, in a **data warehouse** , from the point of view of *data mining* techniques, the *integration by (natural) join* will be very *probably preferred*.
In case of **incomplete information**, specially *missing values*, the usage of the **outer-join** (for instance *left* or *right*) may be useful, but the discussion on this is outside the scope of this report.
We will illustrate the occurrence of the inconsistencies in the integration by joining relations in the next example:

**Example 6**

| $R_1$ | |
|---|---|
| **Husband** | **Wife** |
| Joseph | Mary |

| $R_2$ | |
|---|---|
| **Mother** | **Child** |
| Mary | Jesus |

| $R = R_1 *_{Wife=Mother} R_2$ | | |
|---|---|---|
| **Husband** | **Wife** | **Child** |
| Joseph | Mary | Jesus |

Again, as in the case of the union, even in this very simple example, without any further supplementary information it is impossible to decide whether an inconsistency appeared in the process of the integration of databases.
The comparison of this join with the $\pi$ - union of the same relations:

| $R = R_1 \cup_\pi R_2$ | |
|---|---|
| **Man** | **Woman** |
| Jesus | Mary |
| Joseph | Mary |

shows that the integration by joins against the integration by unions:

- allows **new** *relationships between objects* (entities or their attributes, and this exactly what is usually one looking for in any **data mining** technique), which
- can be the source of **new** *inconsistencies* (having for arguments some of such new relationships) in addition to the inconsistencies known from the unions.

In any case the **IFAR** *methodology* can be used again.

*Remark 7* What was said about the **importance** of the *semantical justification* for the $\pi$ - union holds *even more* for the **joins** as the only condition on $p$ relations $R^{i_k}{}_{q_k}$ to be joinable is:

**A6:** $$\bigcap_{k=1}^{p} D^{i_k}{}_{q_k} ( \pi^{i_k}{}_{q_k} ( B^{i_k}{}_{q_k} ) ) \neq \emptyset \qquad \text{where} \qquad ( \forall\, k \in \widehat{p} ) ( B^{i_k}{}_{q_k} \subset A^{i_k}{}_{q_k} )$$

which is equal to the condition **A5** with a unique difference that $B^{i_k}{}_{q_k} \subset A^{i_k}{}_{q_k}$ and so one can have in principle up to $\prod_{k=1}^{p} ( \sum_{m=1}^{|B^{i_k}{}_{q_k}|} \binom{|A^{i_k}{}_{q_k}|}{m} )$ possibilities of performing the join of $p$ relations (here again a remark similar to the *Remark 6* applies).

# 6 Inconsistencies classification

Now we will suppose the following assumption is valid:

**A7:** *Let* $B_k$ *be* $m$ *(*$m \geq 2$*) databases one wants to integrate,*
$\Sigma_k$ *be* $m$ *corresponding sets of integrity constraints and*
$\Sigma_{m+1}$ *be the set of the integrity constraints corresponding to the result of database integration operation* $\int_{k=1}^{m} \mathcal{B}_k$ such that $\Sigma = \bigwedge_{k=1}^{m+1} \Sigma_k$ is (logically) consistent.

**Definition 5** Let $\mathcal{B}_k$ be $m$ $(m \geq 2)$ databases satisfying assumption **A7**.
We will call any inconsistencies in the result of the database integration $\int_{k=1}^{m} \mathcal{B}_k$ the **database integration inconsistencies**, specially:

- **universe of discourse inconsistencies** $\Leftrightarrow (\exists\, k \in \widehat{m})\,(\exists\, \widetilde{A^i}_k \neq A^i{}_k)$
- **data inconsistencies** $\Leftrightarrow (\exists\, k \in \widehat{m})\,(\exists\, \widetilde{R^i}_k \neq R^i{}_k)$
- **integrity constraints inconsistencies** $\Leftrightarrow (\exists\, k \in \widehat{m+1})\,(\widetilde{\Sigma_k} \neq \Sigma_k)$
- **semantical inconsistencies** $\Leftrightarrow (\exists\, k \in \widehat{m})\,(\exists\, \pi^i{}_k \neq Identity)$

(where $\widetilde{A}$ is the subset of the set $A$ containing no wrong attributes).

*Convention 1* We will call the database integration inconsistencies shortly **integration inconsistencies**.

**Definition 6** The *universe of discourse inconsistencies* and the *integrity constraints inconsistencies* will be called the **conceptual inconsistencies**.

Every type of the integration inconsistencies in our *classification* originates from *different* **sources** and therefore *can be best* **eliminated**, or *at least* **minimized**, at *different* **stages** of the integration of the concerned databases:

- the *conceptual inconsistencies* at the stage of the **schema integration**
(and so any occurrence of these inconsistencies can be a *signal*, in the case of a data warehouse, to *redesign* the **logical schema** of the given data warehouse, or even to think about a *new* **conceptual model** of the data warehouse)
- the *semantical inconsistencies* by **well-considered choice** of the *attribute(s) over which one wants to integrate the databases*, maybe for the purpose of the *envisaged* **data mining** in a given data warehouse
- the *data inconsistencies* by *thorough* **verification** and **validation**, at the *data entry* stage.

*Convention 2* In the following we will use the notation:

| | |
|---|---|
| $\delta$ **- inconsistencies** : | *database integration inconsistencies* |
| **u - inconsistencies** : | *universe of discourse inconsistencies* |
| **d - inconsistencies** : | *data inconsistencies* |
| **i - inconsistencies** : | *integrity constraints inconsistencies* |
| **s - inconsistencies** : | *semantical inconsistencies* |
| **c - inconsistencies** : | *conceptual inconsistencies.* |

# 7 Existence Conditions for the Integration Inconsistencies

We have seen in previous sections that the integration of (relational) databases may lead to inconsistencies.

*In order to eliminate, as much as possible, the occurrences of these inconsistencies* (for instance in a **data warehouse** one wants to build up) *one should try to*, especially in the case of the validity of the following conditions **A1 & A2 & A7 &**

- **A4:** *clear the databases* to be integrated from:
  - **inconsistent data** which *can lead to the* **d** - *inconsistencies*
  - **wrong integrity constraints** which *can lead to the* **i** - *inconsistencies*
  - **wrong attributes** which *can lead to the* **u** - *inconsistencies*
- **A5:** *semantically* **deeply analyze** the *corresponding attributes* in the relations to be integrated by $\pi$ **- unions** to eliminate the **s** - *inconsistencies*
- **A6:** *semantically* **deeply analyze** the *corresponding attributes* in the relations to be integrated by **joins** to eliminate the **s** - *inconsistencies*.

# 8 RIFAR procedure

We will now refine the **IFAR** *methodology* into the following **RIFAR** *procedure*:

STEP 0: **R***esolve the conflicts in* $\Sigma = \bigwedge_{k=1}^{m+1} \Sigma_k$

*Put* $i = 1$

STEP 1: *While* $i < m-1$: *Put* $i = i+1$ & **I***ntegrate the database* $\mathcal{B}_i$ *with* $\int_{k=1}^{i-1} \mathcal{B}_k$

*Put* $j = 0$

SUBSTEP 1A: *While* $j < (k_i - 1)$: *Put* $j = j+1$ & *Integrate the relation* $R^i{}_j$ *with* $\int_{s=1}^{j-1} R^i{}_s \int_{k=1}^{i-1} \mathcal{B}_k$

SUBSUBSTEP 1A2: **F***or every tuple* $t$ *from* $R^i{}_j$ *verify if it does lead to an inconsistency (with respect to the given set of the integrity constraints* $\Sigma_{m+1}$*)*

SUBSUBSUBSTEP 1A2A: *If it does:*
- *remove the corresponding tuple(s) from* $\int_{s=1}^{j-1} R^i{}_s \int_{k=1}^{i-1} \mathcal{B}_k$ *if this does not violate* $\Sigma_{m+1}$, *otherwise make a copy of it/them*
- *put it/them together with* $t$ *into* $\mathcal{I}(\int_{k=1}^{m} \mathcal{B}_k)$
- *index them all by the corresponding integrity constraint(s)*

SUBSUBSUBSTEP 1A2B: *If it does not, integrate it with* $\int_{s=1}^{j-1} R^i{}_s \int_{k=1}^{i-1} \mathcal{B}_k$

STEP 3: **A***nalyze the set* $\mathcal{I}(\int_{k=1}^{m} \mathcal{B}_k)$ *by*:

SUBSTEP 3A: *Decomposing it into subsets indexed by the set(s)* $Q$ *of the same integrity constraint(s)* $\mathcal{I}(\int_{k=1}^{m} \mathcal{B}_k)_Q$

*to find*:
- $\mathcal{I}_{\Sigma_{m+1}}(\mathcal{B})$
- *wrong integrity constraints*
- *wrong descriptions* of data

STEP 4: **R***esolution of inconsistent and wrong "items":*
- *correction of data* (in order to obtain: $\widetilde{R^i{}_j}$)
- *correction of integrity constraints* (in order to obtain: $\widetilde{\Sigma_i}$) and
- *correction of attributes* (in order to obtain: $\widetilde{A^i{}_j}$).

# 9 Conclusion

By *analyzing* some simple examples we have arrived at the **sources** of *possible inconsistencies* when integrating databases (e.g. into a *data warehouse*) and we have *proposed* **classification** of these inconsistencies based on the their sources.

We have find *four* conditions **A3**, **A4**, **A5** and **A6** which can lead to different types of inconsistencies in the process of the integration of databases.
The conditions **A3**, **A4** and **A5** apply to the *integration by unions* while the condition **A6** applies to the *integrations by joins*.

The *occurrence of any (type of the) inconsistencies can*, according to our present knowledge, *provide an useful* **feedback** to, at least, the **conceptual modelling** of a *data warehouse*, to its *logical schema design*, to a more *intelligent* **data entry**, *verification* and *validation*, and to a *better* **selection** of the appropriate **data mining techniques / methods**.
For instance, the occurrence of any of **c - inconsistencies** can trigger a positive feedback to the conceptual modelling of a concrete data warehouse as, depending on its precise type, it can either signal wrong attribute(s) existence in the case of the **u - inconsistencies**, either wrong integrity constraint(s) existence in the case of the **i - inconsistencies**.

As in the case of relations in the relational data model the semantics (meta-data) of data (description of attributes, corresponding integrity constraints, etc.) are – or can be – stored in the same type of relations (so called *system* relations), our **IFAR** *methodology* and **RIFAR** *procedure* can be applied to any conflict of similar schema structures [17] (*value–to–value conflicts*, *attribute–to–attribute conflicts* and *table–to–table conflicts*) on the one side, but also to any conflicts of different schema structures (*value–to–attribute conflicts*, *value–to–table conflicts* and *attribute–to–table conflicts*) on the other side.

The ideas presented here (especially the **RIFAR** *procedure*) have been implemented in a prototype system to provide *support for the resolution of the inconsistencies in the process of the integration of databases* (in general – not only for the *data warehousing* case).

In the future we would like to further elaborate *methodology* and *procedure* by incorporating it in to intelligent agent system and taking more advantage of the soft computing paradigm.

# Bibliography


[1] P. Hájek, T. Havránek: "Mechanizing Hypothesis Formation (Mathematical Foundations for a General Theory)," *Springer–Verlag* (1978), 396 pp.

[2] P. Hájek, I. M. Havel, and M. Chytil: "The GUHA method of automatic hypotheses determination," *Computing* **1** (1966), pp. 293–308.

[3] C. Batini, M. Lenzerini: "A methodology for data schema integration in the entity relationship model," *IEEE Transaction on Software Engineerings***10** (6), 1984, pp. 650–664.

[4] C. Batini, M. Lenzerini, and S. B. Navathe: "A comparative analysis of methodologies for database schema integration," *ACM Computing Surveys* **18** (4), 1986, pp. 323–364.

[5] M. Garcia-Solaco, F. Saltor, M. Castellanos: "A Structure based Schema Integration Methodology," in *Proceedings of 11th IEEE International Conference on Data Engineering* (**ICDE-95**), 1995, 505-512

[6] V. Ramesh and S. Ram: "Integrity constraint integration in heterogeneous databases: An enhanced methodology for schema integration," *Information Systems* **22** (8), 1997, pp. 423–446.

[7] G. Santucci: "Semantic schema refinements for multilevel schema integration," *Data & Knowledge Engineering* **25** (1998), pp. 301–326.

[8] F. S. C. Tseng, J.-J. Chiang, and W.-P. Yang: "Integration of relations with conflicting schema structures in hererogeneous database systems," *Data & Knowledge Engineering* **27** (1998), pp. 231–248.

[9] A. Poulovassilis and P. Mc.Brien: "A general formal framework for schema transformation," *Data & Knowledge Engineering* **28** (1998), pp. 47–71.

[10] I. Kwan and J. Fong: "Schema integration methodology and its verification by use of information capacity," *Information Systems* **24** (5), 1999, pp. 355–376.

[11] L. Palopoli, L. Pontieri, G. Terracina and D. Ursino: "Intensional and extensional integration and abstraction of heterogeneous databases ," *Data & Knowledge Engineering* **27** (1998), pp. 231–248.

[12] A. Gupta, H. V. Jagadish and I. S. Mumick: "Data Integration Using Self-Maintainable Views ," in *Proceedings of the 5th International Conference on Extending Data Base Technology* **EDBT-96**, Lecture Notes in Computer Science – *LNCS* **1057**, Springer-Verlag, 1996, pp. 140–144.

[13] J. D. Ullman: "Information integration using logical views," *Theoretical Computer Science* **239** (2000), pp. 189–210.

[14] Y. Arens, C.-Y. Chee, C.-N. Hsu and C. A. Knoblock: "Retrieving and Integrating Data from multiple information Sources," *International Journal of Intelligent and Cooperative Information Systems* **2** (2), 1993, pp. 127–158.

[15] M. Arenas, L. E. Bertossi and J. Chomicki: "Consistent Query Answers in Inconsistent Databases," in *Proceedings of the Eighteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, May 31 - June 2, 1999, Philadelphia, Pennsylvania* (1999), pp. 68-79.

[16] R. Hull and G. Zhou: "A framework for supporting data integration using the materialized and virtual approaches," in *Proceedings of the ACM SIGMOD International Conference on Management of Data* (1996), pp. 481–492.

[17] M. E. Orlowska, H. Li, and C. Liu: "On integration of relational and object-oriented database systems," in *LNCS* **1338** *(Proceedings of SOFSEM'97)*, Springer-Verlag, 1997, pp. 294–312.

[18] C. J. Date: "An Introduction to DATABASE SYSTEMS," *(seventh edition)*, *Addison-Wesley (Longman, Inc.)*, Reading, Massachusetts 2000.

[19] C. Lee, C.-J. Chen, and H. Lu: "An aspect of query optimization in multidatabase systems," *ACM SIGMOD Record* **24** (3), 1995, pp. 28–33.

[20] A. Hunter: "Moerging potentially inconsistent items of structured text ," *Data & Knowledge Engineering* **34** (2000), pp. 305–332.

[21] J. L. Castro and J. M. Zurita: "A heuristic in rules based systems for searching of inconsistecies," *Journal of Information Sciences* **108** (1998), pp. 135–148.

[22] M. Chen, Q. Zhu, Z. Chen: "An integrated interactive envirenment for knowledge discovery from heterogeneous data resources ," *Information & Software technolog* **43** (2001), pp. 487–496.

[23] H. Wang: "A Survey of Mathematical logic," *Science Press*, Peking 2000.

[24] O. Etzion and B. Dahav: "Patterns of self-stabilization in database consistency maintenance," *Data & Knowledge Engineering* **28** (1998), pp. 299–319.

[25] G. Pernul, A. M. Tjoa, W. Winiwarter: "Modelling data secrecy and integrity," *Data & Knowledge Engineering* **26** (1998), pp. 291–308.

[26] E. O. de Brock: "A general tratmnt of dynamic integrity constraints," *Data & Knowledge Engineering* **32** (2000), pp. 223–246.

[27] M. Zviran and C. Glezer: "Towards generating a data integrity standard," *Data & Knowledge Engineering* **32** (2000), pp. 291–313.

[28] J. S. Robbins, A. C. Stylianou: "Post-merger systems integration: the impact on IS capabilities ," *Information & Management* **36** (1999), pp. 205–212

# 10 Appendix

*Notation 1* $\widehat{m} = \{1, 2, \cdots, m\}$ $\qquad (\widehat{0} = \emptyset)$

**Definition** A **relation** in the *relational model of data* (RMD) is any triple $\langle A, D, T \rangle$ where:

1. $A$ is a finite set of **attribute names**.
2. $D$ is a mapping which maps every attribute name $a \in A$ to a (countably infinite) set noted $D(a)$ and called the **domain** *of the attribute* $a$.
   Let us: *denote* by $D(A)$ the *union* of all $D(a)$ and
   *call* it the **underlying database domain**.
3. $T$ is a finite set of **mappings** $t$ from $A$ to the underlying database domain such that $t(a) \in D(a)$ for all $a \in A$.

*Notation 2* The *cardinality* of the set $A$ will be denoted by $|A|$.

*Notation 3* $T[A_1] = \{ t\colon A_1 \to D_1(A_1) \mid (\exists u \in T)(t(A_1) = u(A_1)) \}$

**Definition** Let $R = \langle A, D, T \rangle$ be a relation and $A_1 \subset A$.
The **projection** of the relation $R$ over $A_1$ is the relation noted
$R[A_1] = \langle A_1, D_1, T_1 \rangle$ such that:

1. $D_1 = D/A_1$ (the *restriction* of the mapping $D$ on the subset $A_1$ of $A$)
2. $T_1 = T[A_1]$

*Notation 4* Let $A_i = \{ a_{ij} \mid j \in \widehat{|A_i|} \}$, $i \in \{1, 2\}$.

$(\forall j \in \widehat{|A_i|})\ (D_1(a_{1j}) \cap D_2(a_{2\pi(j)}) \neq \emptyset)$

($\pi$ being an appropriate permutation in $\widehat{|A_i|}$)

$\Updownarrow$

$D_1(A_1) \cap D_2(\pi(A_2)) \neq \emptyset$

**Lemma** $(D_1(A_1) \cap D_2(\pi(A_2)) \neq \emptyset) \Rightarrow (|A_1| = |A_2|)$

**Definition** Let $R_i = \langle A_i, D_i, T_i \rangle$, $i \in \{1, 2\}$, be two relations such that:

1. $(\exists A_{21} \subset A_2)\ (|A_1| = |A_{21}|)$
2. $D_1(A_1) \cap (D_2/A_{21})(\pi(A_{21})) \neq \emptyset)$
   ($\pi$ being an appropriate permutation)
3. $T_1(A_1) \subset T_2[A_{21}](\pi(A_{21}))$

Then we will say that the relation $R_1$ is a **subrelation** of the relation $R_2$
– what we will note by $R_1 \subset R_2$.

**Definition** A **relation schema** in the RMD is any couple $\langle A, D \rangle$ with:

1. $A$ being a (finite) set of attribute names.
2. $D$ being a mapping which maps every attribute name $a \in A$ to the corresponding domain $D(a)$ of the attribute $a$.

**Definition** Let $R = \langle A_1, D_1, T \rangle$ be a relation in the RMD and
$\mathcal{S} = \langle A, D \rangle$ be a relation schema in the RMD such that:

1. $A_1 = A$
2. $D_1 = D$.

Then we will say the relation $R$ is **defined over the relation schema** $\mathcal{S}$ and we will note it: $R \sqsubset \mathcal{S}$.

**Definition** A **database schema** $\mathcal{D}$ in the RMD is any finite set of relational schemata $\mathcal{S}_i$ :
$\mathcal{D} = \{ \mathcal{S}_i \mid i \in \widehat{m} \}$.

*Notation 5* $\bigcup_{i=1}^{m} T_i(\pi_i(A_i)) = \{ t\colon A \to D(A) \mid (\exists i \in \widehat{m})\ (\exists u_i \in T_i)$
$(u_i(\pi_i(A_i)) = t(A)) \}$

**Definition** Let $R_i = \langle A_i, D_i, T_i \rangle$ be $m$ $(m \geq 2)$ relations.
The $\pi$ **- union** of relations $R_i$ is the relation noted $\bigcup_{\pi}{}_{i=1}^{m} R_i = \langle A, D, T \rangle$ such that:

1. $D(A) \cap (\bigcap_{i=1}^{m} D_i(\pi_i(A_i))) \neq \emptyset$
2. $T = \bigcup_{i=1}^{m} T_i(\pi_i(A_i))$

*Convention 1* In the case of permutations $\pi_i$ being *identities* we will omit the prefix $\pi$ - and speak shortly only about the **union** and note it $\bigcup_{i=1}^{m} R_i$ .

*Notation 6* $D(a_j) = \bigcup_{i=1}^{m} D_i(a_j)$ , $(\forall j \in |\widehat{A}|)$ $\quad (A = \bigcup_{i=1}^{m} A_i)$

$$\Updownarrow$$

$$D = \bigcup_{i=1}^{m} D_i$$

*Notation 7* $*_{\pi_1(B_1)=\pi_i(B_i)} T_i = \{ t\colon A \to D(A) \mid ((\forall i \in \widehat{m})(\exists u_i \in T_i))$
$((t(A_j) = u_j(A_j)) \wedge$
$(u_1(\pi_1(B_1)) = u_i(\pi_i(B_i))))\}$

**Definition** Let $R_i = \langle A_i, D_i, T_i \rangle$ be $m$ $(m \geq 2)$ relations and
$B_i$ be $m$ sets of attributes such that
$((\forall i \in \widehat{m})(B_i \subset A_i)) \wedge (\bigcap_{i=1}^{m} D_i(\pi_i(B_i)) \neq \emptyset)$

The **join** of the relations $R_i$, *according to the attributes sets* $B_i$, *with respect to the equality*, is the relation noted
$*_{\pi_1(B_1)=\pi_i(B_i)} R_i = \langle A, D, T \rangle$ where: 1. $A = \bigcup_{i=1}^{m} A_i$
2. $D = \bigcup_{i=1}^{m} D_i$
3. $T = *_{\pi_1(B_1)=\pi_i(B_i)} T_i$

*Convention 2* In case of permutations $\pi_i$ being identities, the equality of $B_i$ and such that they are *maximal* (in *set inclusion* sense) with such a property, we will omit the *index* $_{\pi_1(B_1)=\pi_i(B_i)}$ by the $*$ and will obtain the *natural join* of $R_i$.