DOMAINS, RELATIONS AND RELIGIOUS WARS

R. Camps. Escola Universitària Politècnica. Vilanova i la Geltrú. 08800 - Spain

"An abstract term is like a valise with a false bottom, you may put in it what ideas you please, and take them out again, without being observed" *Alexis de Toqueville (1835)*

"data models have much in common with religion" *M. Stonebraker (1988)*

1. MOTIVATION

The Third Manifesto [6], written by H.Darwen and C.J.Date, and published in this journal in March 1995, attempts an OO revitalization or reincarnation of the Relational Model, RM. C.J.Date claims [19] that, were the RM fully implemented, and particularly *domains*, it would provide "*all the functionality promised by object-oriented systems, and more*".

Shocked by this statement, I thought it useful to look backwards and review the evolution of the meaning given by the RM world, and particularly by C.J.Date himself, to the term "domain" on which the reincarnation attemp is based. Exercising memory can help understand the present and learn from past errors.

2. LOTS OF RMs

To begin with, which RM do we mean? There are several lines of RM and each has had its own evolution. The original line was originated by E.F.Codd [5] who developed during the Seventies what he later [1] named *RM/V1*. In 1979 he proposed a new model, the *RM/T* [4] that meant a huge change from the original RM approach. In the Eighties, sensing that plain people did not keep up with him, Codd wrote [1]:

> *"Vendors of DBMS products have in many cases failed to understand the first version RM/V1, let alone RM/T."* Therefore the RM/T will be *"gradually dropped into the sequence of versions RM/V2, RM/V3,.."*

In 1985 Codd defined in the popular journal *ComputerWorld* [2] a set of 12+30 rules that a DBMS must comply with in order to qualify as "fully relational." Later, in 1990, Codd [1] put forward the *RM/V2*, constituted by 333 features.

From the mid Seventies, Date devoted himself to spread and clarify the RM concepts as defined by Codd. But in succesive editions of his famous book *"An Introduction to Database Systems"* [11] he slowly evolved in disagreement with Codd. Date's last vision of the RM is as defined in *The Third Manifesto*.

3. FROM 1FN TO OO

One of the strongest points of Object Oriented DBMSs as compared to Relational DBMSs is, that they allow a more natural modeling owing to their ability to define complex structures, while in RDBMSs only elementary tables can be used. Why do most RDBMSs support only elementary structures? The answer, according to Date, is not that the RM were based precisely on this structural simplicity, but that DBMS vendors never understood the RM. Today, Date's claim [6] is that there always has been room for complex structures in the RM.

In fact, an essential pillar on which the RM was built, the basis for the simplicity of its data structures, is the requirement of the first normal form, 1NF, which used to be stated as requiring domains to consist only of atomic, non-decomposable, values. To the users of pre-relational DBMSs accessed from COBOL, PL/I, etc, it was one of the most surprising, and most difficult to accept, of the RM principles. They viewed this requirement as a strong restriction and a step backwards. The subject was a major battlefield in the confrontation between DBTG-CODASYL and RM supporters. But the RDBMSs coming up in the market, and the

discourse of the *official* relational literature, made, until the turn of our decade, a strict requirement of this basic restriction. Compelled by the *fear of Codd*, the mere mortals (the industry and the users) reluctantly accepted the *Law*, although in some applications the decision was to go on with the pre-relational DBMSs, or lately, to switch to the new OODBMSs. You must have a weak memory, or be young, to claim that the need for complex columns is a novelty and that traditional "data processing" applications require only atomic columns.

Since the late Seventies, some researchers have been proposing a significant extension to the RM which would allow tuple components to be relations. This would allow the representation of complex structures. This extended RM is usually known as NF2 (Non First Normal Form) or Nested-Relational. The major outcome of this line of research is non-1NF normalization theory [12] [13] [14]. The supporters of the classical RM, like Date, do not espouse this "heretical" model, as we will see below.

Nowadays, ANSI/SQL-92, and thus most current RDBMSs, still reject non-simple columns (except for the *date* type). This situation is about to change with the advent of Object-Relational DBs [15] that are the response from the relational DB vendors to the OO wave. As Stonebraker said [16]

> *"the CODASYL vendors were asleep when the relational wave rolled over them. The current commercial [relational] vendors remember this clearly and are not asleep"*

4. DOMAINS: THE SEVENTIES AND EIGHTIES

In Codd's original paper [5] from **1970**, it can be read that

> - "*simple domains --domains whose elements are atomic (non-decomposable) values*."
> - "*the terms attribute and repeating group in present data base terminology are roughly analogous to simple domain and non-simple domain, respectively*"
> - "*domains must be uniquely identifiable, without using position*"

Apart from the confusion created by using *domain* with the sense of *attribute* or *column*, the meaning of the text seems quite understandable. Furthermore, Codd said that, in the mathematical theory of relations, non-atomic domains were acceptable, and thus a domain could have relations among its elements. However, a relation with non-simple elements would not comply with the normalization condition, which was a requirement for relations in a database.

In the first edition (**1975**) of his book [11], Date, taking up from Codd's text, wrote that

> "*Every value within a relation --i.e., each domain value in each tuple-- is an atomic (non-decomposable) data item (e.g., a number or a character string)*."

Nowhere in the book was any example with a domain more complex than *number* or *character string*; there was not even any example using *dates*.

In the third edition (**1981**), the sentence quoted above underwent slight changes. Date substituted "*attribute*" for "*domain,"* blaming Codd for the confusion. After "*non-decomposable,*" he added "*so far as the system is concerned*" without further comments. Nowhere in the book was any hint of any other kind of domain than those, all simple, in the first edition. Moreover, in this 3rd edition it was said that "*relations [...] are defined over simple domains (domains that contain atomic values)*."

Those were years of research and consolidation of an adequate terminology. It was also the time when the first DB war began, a bitter fight between the relational camp and the DBTG-CODASYL camp [16]. The relational side, champions of the *Truth*, used the principles/dogmas of the RM as weapons against the enemy. The relational camp was mainly composed by IBM (maneuvering against a possible non-IBM industry standard) and the academic world although Codd had a hard time in IBM before his ideas were accepted [7]. In the DBTG camp was most of the rest of the professional world. Under the cover of technical and scientific arguments, relevant commercial interests were being disputed.

Date published in **1984** [10] a paper to disprove some *myths* on the RM. He explained that a simple domain is associated with a simple (or atomic) data type, and that the RM also allows composite domains. He gave an example of the latter, and suggested, on the fly, a syntax (using PL/I or COBOL style levels) to describe it and to deal with these composite domains/columns.

On the simple types, he said that they may in fact have an internal structure,

> "*the DBMS would not have any knowledge of that internal structure, but the user would, and could exploit that knowledge in a variety of ways --[e.g.:] manipulating substrings. [...] . I do not intend by the foregoing to suggest that such attempts to subvert the system should be encouraged. [...] Even 'simple' data types may have an internal structure that is known to the system in some cases. Complex numbers are a case in point.*"

> "*There is a need in some applications to deal with a vector or array data type also. [...] However, such vectors and arrays must be of fixed bounds; for otherwise they would in fact be repeating groups, which are definitely not part of the model.*"

> "*[...] the lack of composite field support, in systems if not in the model (which makes dealing with composite data rather clumsy). [...] When faced with difficulties such as these, there are basically two things we can do: 1. We can find a way around the problem within the existing methodology. 2. We can extend the methodology appropriately. [...] in the case of composite fields [...] the second of this is clearly the right thing to do.*"

In **1986**, the 4th edition of Date's book [11] was published. There, value non-decomposability was no longer "*so far the system is concerned,*" but just "*as far as the model is concerned.*" In this edition, there was a long new section devoted to domains. There was no mention of extending the RM to support composite domains, and the only example of them was a *date* column.

> *"Domains of atomic values are more accurately referred to as simple domains, to distinguish them from composite domains. However, domains are invariably assumed to be simple unless explicitly stated to be otherwise."*
>
> *"Relations do not contain repeating groups. A relation satisfying this condition is said to be normalized [...] The relational model does not permit relation-valued domains."*
> *"Do not confuse relation-valued and composite domains."*

As we can see, it was not clear at all from this text whether a relation with composite attributes does or does not comply with the forth property of relations, "*All simple attribute values are atomic.*" Furthermore, in this edition it was said that "*the relational notion of domain is closely related to [the programming language notion of] data type.*"

Date concluded that

> "*Indeed, the domain concept is in fact considerably more complex than it might appear at first sight (which is perhaps why most systems currently do not support it).*"

Would not it be more reasonable to look for the cause in the RM's lack of a type system?

In **1988**, Date wrote a proposal under the title "*A Proposal for Adding Date and Time Support to SQL.*" When republished in 1990 [8], he changed the title to "*Defining Data Types in a Database Language,*" although no change at all was made to the original text.

In texts by Codd before 1979, mention was made of "extended data types". In **1974**, a paper was published signed by Date, although, according to him, written entirely by Codd (republished in [9]). It was said there that "*to enrich the variety of data types available [..] so that it might model the real world more faithfully*" could be useful in a user schema, but not in the principal schema.

In **1981**, F. Saltor [18] said that

> "*[RM/T] is an attempt to incorporate some of the recent research, but not all of it, not even what seems to me the most promising, that on abstract types in databases.*"

On being awarded in **1981** the ACM Turing Prize, Codd said [3] that

> "*Ongoing investigations focus on the incorporation of abstract data types into database languages*"

In **1983**, the INGRES system (a prototype version developed in the University of California by Stonebraker) had already user-definable ADTs (Abstract Data Types) to be used in column descriptions. As early as **1984** [17], the Stonebraker's team was proposing to add inheritance to the ADT in INGRES.

5. DOMAINS: THE NINETIES

When, in **1990**, Codd proposed the RM/V2 [1], he devoted 9 rules to the topic of *data types*. One of them made explicit mention of "*User defined Extended Data Types."*

In **1990**, Date published in [8] a paper titled "*What is a Domain?*" with the intent to make a "*definite statement*" on what domains are about, thus clarifying a subject on which there had been, until then, *"a certain amount of confusion in the existing literature*". He noted that Codd did not make the concept clear until 1988, and added that "*I cannot really agree with [Codd] [...] it fails to do full justice to the [domain] concept*."

Date observed that, in a normalized relation, domains cannot be *relation-valued* because would then be *repeating groups*; however, they can indeed be composite domains (tuples). A section in the paper is devoted to the subject. He wonders, "*But does the atomic value assumption really stand up under close scrutiny?*" and comments:

> *"The existence of composite values is not the only reason to question the atomic value assumption. [...] Several systems include support for [...] built-in data types that are quite definitely <u>not</u> atomic, but rather do have an internal structure that is known to the system, dates and times provide an obvious example [...] Codd himself is on record as agreeing that such facilities might sometimes be desirable."*
>
> *"In fact, it is not clear that there is any <u>fundamental</u> reason why data values need to be atomic" "Indeed, it is widely accepted that, [...] such atomicity becomes something of a millstone."*
>
> *"We see that data value atomicity cannot truly be regarded as a hard and fast requirement. [...] In general, domain values should be allowed to have an internal structure of arbitrary complexity."*

To conclude, he flatly states that *"a domain is nothing more nor less than a <u>user-defined data type</u>*."

It was also in **1990** that the 5th edition of [11] was released. There, Date began to use the term *scalar* in some of the places were *atomic* was used in the 4th edition (*atomic* is no longer used in "*The Third Manifesto*"). The term *scalar* appeared in [8] in a footnote saying that the concept of atomic (non-decomposable) value is called *scalar* in some programming languages such as PL/I.

Date commented there that "*there is nothing to stop a user from defining a given column as <u>character string</u> but using that column (tacitly) to contain structured values of any degree of complexity (e.g., a repeating group). In other words, there is really no way to <u>enforce</u> the atomicity requirement.*" He further made a warning that "*Actually, atomicity of scalar values is a rather slippery concept".* And asked himself, *"Why do we insist on normalization?*" to which he answered "*<u>simplicity</u>*".

In this fifth edition, he made it clear that composite values "*do <u>not</u> present any problem in regard to [the atomic value] assumption*". He also introduced "*component selector functions*" using as examples MONTH, DAY and YEAR.

In **1994**, in an interesting paper to celebrate the silver anniversary of the RM [7], Date claimed that two good ideas from the OO approach should be incorporated into the relational systems:*"<u>User-defined data types</u>*" and *"Inheritance".*

It is surprising to see how Date turns the lack of an adequate definition of domains/types into an implementers' fault.

> "*A relational system that implemented domains properly would be able to do all the*

things that OO advocates claim their systems can do and relational systems cannot."

Early in **1995**, the 6th, and last up to now, edition of [11] was released. There, Date supports bringing the RM and the OO approach nearer one to the other, with almost the same words as in "*The Third Manifesto*". There is a chapter headed "*Toward an OO/Relational Rapprochement*" were he puts forward a way to build a system both relational and OO, with the strong points of both approaches, and none of their drawbacks:

"the OO capabilities represent a very natural extension (actually not an extension at all) to the relational model" "Relational systems can now handle complex application areas such as CAD/CAM."

How can Date claim that this is not an extension to the RM? At least, it is an extension to the spirit and the letter of the RM as presented by him until the late Eighties. The key of the revitalization of the RM are, of course, domains. We can read that,

"*the fundamental construct in relational systems- mostly not implemented, unfortunately, in today's relational products- is the domain"*

"domains (and therefore, relational attributes) can contain absolutely anything -arrays, lists, stacks, documents, photographs, maps, blueprints, etc., etc."

In [11] 6th ed. Date also says

"we can have domains, and therefore attributes, that are relation-valued." "relation-valued attributes [..] -if correctly implemented- do not violate the relational requirement for normalization [..] contrasted [..] with nested-relation support (which does violate that requirement)."

Date, today, considers domains as UDT (User Defined Type) of arbitrary internal complexity, and the fundamental construct in the RM. As for composite domains, he claims that he no longer agrees with Codd that they should be explicitly included in the RM.

Looking at all the quotations I have made from [11] and considering that this book was the *bible* where most university graduates all over the world learnt, I believe that Date can be held partly responsible for the lack of implementation of domains (domains viewed as UDT). But he states

"Indeed, an argument can be made that the whole reason that OO systems look attractive is precisely because the relational vendors to date have failed to support the relational model adequately"

The Third Manifesto [6], published in March 1995, consists of the same ideas in 6th ed. of [11], but presented in a more structured and formalised way. Darwen and Date claim they do not feel comfortable with the word *Manifesto* in the title, because they think their paper not to be a statement of intent or opinion, but "*a matter of science and logic*."

In the Manifesto, the domain is raised to the highest place in the RM. The first of the 33 prescriptions, RM1, coming, according to Darwen and Date, from the RM, refers to domains and, among other things, states that,

"*A **domain** is a named set of values. Such values [...] shall be of arbitrary complexity [...] we explicitly permit scalar values to be arbitrarily complex; thus, e.g., an array of stacks of lists of... (etc.) might be regarded as a scalar value in suitable circumstances*"

In prescriptions RM6 and RM7, also coming from the RM according to the authors, it is said that,

"*it shall be possible to define a domain whose values are tuples*"
"*it shall be possible to define a domain whose values are relations*"

They still call such domains *scalars* (because they place themselves in an abstraction level higher than the detailed domain definition), and thus they feel able to claim that they are not making any amendment to the classic relational theory, and that they still require 1NF: *"We explicitly do not espouse NF2 [...] which involves major extensions to the classical Relational Algebra"*

In the eighth proscription belonging to the RM, the authors say that adequate data language

> "*[..] shall not include any specific support for composite domains or composite columns [...] since such functionality can be achieved if desired through the domain support already prescribed*"

This is to say, only scalar values are accepted, but a scalar value may be arbitrarily complex, and therefore even repeating groups are acceptable.

6. FINAL COMMENTS

The discourse on atomicity and 1NF was often a fundamentalist discourse, strongly opposing the complex columns in the pre-relational systems. This battle was part of the first database war. At least, this was the losers understanding, and that of general people, who, in their vast majority, reluctantly accepted the dictates of *The Law*.

With the benefit of hindsight we can imagine that, had the religious war of the late Seventies between DBTG and relational camps been a rational debate, free of fanaticism and commercial interests, today's scenario would be quite different. Perhaps as early as the early Eighties , most commercial DBMSs would have had not only a standard query language based on relational algebra/calculus, but also support for complex user defined data types and a "storage schema" standard language including the definition of links to implement "precomputed joins".

It would be good to exercise the memory and learn from past errors made by both sides. This is not easy as it requires a self-critical approach, a not very common attitude.

Indeed, people as relevant in the DB history as Codd and Date, have earned the right to manifest their opinions, argue, rectify and evolve. In [11] (6th ed.) Date admits he has changed his mind many times and, assisted by Bertrand Russell, he reminds us that scientific research inevitably leads to changing opinions. In short, a dogmatic spirit does not contribute much to progress, and it is always profitable to review our old opinions and beliefs.

REFERENCES

1 E. F. Codd: *The Relational Model for Database Management Version 2*. Reading, Mass.: Addison-Wesley (1990).

2 E. F. Codd: "Is your DBMS really relational?" (1st part), and "Does your DBMS run by the rules?" (2nd part), *ComputerWorld* (14th and 21st October 1985).

3 E. F. Codd: "The 1981 ACM Turing Award Lecture "Relational Database: A Practical Foundation for Productivity"", *Comm ACM 25*, 2 (February 1982).

4 E. F. Codd: "Extending the Database Relational Model to Capture More Meaning", *ACM TODS 4*, 4 (December 1979).

5 E. F. Codd: "A Relational Model of Data for Large Shared Data Banks", *CACM 13*, 6 (June 1970).

6 H. Darwen and C. J. Date: "The Third Manifesto", *ACM SIGMOD Record 24*, 1 (March 1995)

7 C. J. Date: "Many Happy Returns!", *Database Programming & Design 7*, 9 (September 1994).

8 C. J. Date, *Relational Database Writings 1985-1989*. Reading, Mass.: Addison-Wesley (1990).

9 C. J. Date: *Relational Databases: Selected Writings 1974-1985*, Addison-Wesley (1986).

10 C. J. Date: "Some Relational Myths Exploded", *InfoIMS 4*, 2 and 3 (1984). Republished in [9]

11 C. J. Date: *An Introduction to Database Systems*. Reading, Mass.: Addison-Wesley. 1st ed. 1975 (Volume I) 6th ed. 1995 (in a single volume)

12 A. Makinouchi: "A Consideration on Normal Form of Not-Necessarily-Normalized Relations in the Relational Data Model",

Proc. 3rd International Conference on Very Large Data Bases (1977)

13 W. Y. Mok, Yiu-Kai Ng, and D. W. Embley: "A Normal Form for Precisely Characterizing Redundancy in Nested Relations", *ACM TODS 21*, 1 (March 1996)

14 M. A. Roth, H. F. Korth, and A. Silberschatz: "Extended algebra and calculus for nested relational databases", *ACM TODS 13,* 4 (Dec.1988)

15 M. Stonebraker: *Object-Relational DBMSs*. Morgan Kaufmann (1996)

16 M. Stonebraker: *Readings in Database Systems*. Morgan Kaufmann (1994)

17 J. Ong, Fogg, and Stonebraker: "Implementation of Data Abstraction in the Relational System INGRES", *ACM SIGMOD Record* (March 1984).

18 F. Saltor: "Reflexió crítica sobre RM/T", Convención Informática Latina (CIL) Barcelona 1981.

19 "Objects and SQL: Strange Relations?", Proceedings of the 1995 ACM-SIGMOD, *ACM SIGMOD Record 24*, 2 (June 1995).