ELSEVIER

Computers & Geosciences 32 (2006) 1368–1377

COMPUTERS & GEOSCIENCES

www.elsevier.com/locate/cageo

The problem of missing data in geoscience databases

Stephen Henley*

Resources Computing International Ltd., 185 Starkholmes Road, Matlock, Derbyshire DE4 5JA, UK

Received 14 September 2005; received in revised form 5 December 2005; accepted 16 December 2005

Abstract

SQL is the (more or less) standardised language that is used by the majority of commercial database management systems. However, it is seriously flawed, as has been documented in detail by Date, Darwen, Pascal, and others. One of the most serious problems with SQL is the way it handles missing data. It uses a special value 'NULL' to represent data items whose value is not known. This can have a variety of meanings in different circumstances (such as 'inapplicable' or 'unknown'). The SQL language also allows an 'unknown' truth value in logical expressions. The resulting incomplete three-valued logic leads to inconsistencies in data handling within relational database management systems. Relational database theorists advocate that a strict two-valued logic (true/false) be used instead, with prohibition of the use of NULL, and justify this stance by assertion that it is a true representation of the 'real world'. Nevertheless, in real geoscience data there is a complete gradation between exact values and missing data: for example, geochemical analyses are inexact (and the uncertainty should be recorded); the precision of numeric or textual data may also be expressed qualitatively by terms such as 'approximately' or 'possibly'. Furthermore, some data are by their nature incomplete: for example, where samples could not be collected or measurements could not be taken because of inaccessibility.

It is proposed in this paper that the best way to handle such data sets is to replace the closed-world assumption and its concomitant strict two-valued logic, upon which the present relational database model is based, by the open-world assumption which allows for other logical values in addition to the extremes of 'true' and 'false'. Possible frameworks for such a system are explored, and could use Codd's 'marks', Darwen's approach (recording the status of information known about each data item), or other approaches such as fuzzy logic.

Keywords: Relational database; Open-world assumption; Closed-world assumption; Missing data; SQL; Logic; Fuzzy logic

1. Introduction

> Reports that say that something hasn't happened are always interesting to me, because as we know, there are known knowns; there are things we know we know. We also know there are known unknowns; that is to say we know there are some things we do not know. But there are also unknown unknowns—the ones we don't know we don't know—Donald Rumsfeld, US Secretary of Defense

In the geosciences (as in other observational sciences, and as in military intelligence), it is very common for observational data sets to be incomplete. Data items may be missing altogether, or they may be imprecise in one way or another. There are many things we do not know—and many more that we do not know we do not know.

*Tel.: +44 1629 581454.
E-mail address: stephen.henley@resourcescomputing.com.

0098-3004/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cageo.2005.12.008

If the data are tabulated, then whether a database management system is used or not, some placeholder must be used to indicate the absence of a data item. In systems which are based on SQL, this placeholder is known as 'NULL' and conventionally represented by an empty character string. However, from the earliest days of relational databases it has been recognised that there can be logical problems in handling relations in which there are NULLs. Indeed, relational database theorists (for example, Date, 1995, 2005; Date and Darwen, 1998, 2000; Pascal[1]) have insisted for many years that a table which contains NULLs is not even a relation, because they assert that the predicate (the logical statement), corresponding to any tuple (row) in which there are NULLs, has no meaning.

In order to deal with missing data, Pascal (see footnote 1) advocates an absolutist approach which eliminates the use of 'NULL' altogether. His solution is to use a table containing the data (as a user I/O medium—not constituting any part of the database)—with nothing at all defined for data items which are missing, but rather a separate audit trail table listing the tuples and attributes which are missing. The main data table, as long as it contains 'holes' (i.e. NULLs) is not accepted as a relation but is merely a table, from which (using the audit trail data) a set of valid relations can be created by partitioning the table into a number of sub-tables—each of which is a valid relation because it does not contain any nulls. Unfortunately, the number of relations which can be required increases steeply (2^m) with the number m of attributes that can contain missing values. If it is required to carry out operations such as a relational join on two or more such tables, the number of relations needed can soon become astronomical. Pascal dismisses this complexity as something which can be hidden from the user and automated within the database management system implementation, although of course the problem still remains, even if hidden.

However, it seems that the complexities of this approach may be unnecessary if the predicate logic proposition for a tuple is modified a little. For example, the example proposition which he presents (p. 10):

> Employee uniquely identified by employee number (EMP#) has name (ENAME) works in department (DEPT#), was hired on (HIREDATE), earns salary (SALARY).

could be replaced by

> Employee uniquely identified by employee number (EMP#) has name (ENAME) *is reported to work* in department (DEPT#), *is reported to have been* hired on (HIREDATE), *is reported to earn* salary (SALARY).

As we shall see later, this re-formulation is very important. The predicate logic used in relational databases should be a set of statements about our knowledge of the world rather than statements of the 'absolute truth' which is usually unknowable. Indeed, if they were statements about 'absolute truth', then this would invalidate the use of relational databases for all scientific applications—as well as for military intelligence and many business applications.

In the above example, if (SALARY) is unknown, and represented by 'NULL', then the new formulation is still true and still records a valid and meaningful fact in its entirety: '…is reported to earn a salary the actual value of which we do not know at present'.

2. Darwen's proposal

Another way that has been proposed to circumvent the logical problems associated with representing missing data was suggested by Darwen.[2] In Darwen's example, although he reaches this by a roundabout route involving horizontal and vertical decompositions of the original relation, addition of extra attributes, and subsequent recombination, the effect is that a 'salary' attribute is replaced by an attribute of 'sal_info' containing *information about* salaries—which will be either the salary itself if known, or words such as 'salary unknown' if unknown or 'unsalaried' if inapplicable. The type of such an attribute must be defined to allow both the full range of possible actual values AND the full set of descriptors that must be used in the absence of an actual value. This is similar to the solution proposed above. Such an interpretation of the data in a table may allow it to retain its full relational credentials, but of course this is at the cost of requiring that either applications or the database

[1]Pascal, F: The Final Null in the Coffin, 2004, http://www.dbdebunk.com

[2]http://www.TheThirdManifesto.com: How to handle missing information without using nulls: presented at Warwick University, 9 May 2003.