MISSING AND INAPPLICABLE VALUES

I would like to comment on E. F. Codd's article on missing database values [1]. As a database administrator (DBA) working in industry, I frequently encounter examples of missing and inapplicable database values. A large subset of these cases can be explained rather simply. They are the result of not placing attributes at the appropriate entity levels in generalization hierarchies. An analysis of Codd's example of inapplicable sales commission database values [1] provides an illustration of this. Following the conventions of Codd's RM/T [2], a salesman entity type can be defined as a subtype of the employee entity type, per the category job type. Since only salesmen have sales commissions, the attribute should be associated with a relation scheme representing the salesman entity type rather than the employee entity type. The relation scheme representing the employee entity type would not have the sales commission attribute and thus an employee base relation would not have tuples with missing and inapplicable sales commission database values. A derived relation, on the other hand, might have tuples with missing and inapplicable database values. For instance, the outer natural join of the employee and salesman relations would potentially contain tuples representing employees who are not salesmen and thus potentially contain missing and inapplicable database values.

The concepts underlying the above analysis are well-known (see [2, 3]). Codd [2] gives the following reason for separating the members of a generalization hierarchy into separate entity types:

> We do this only if different kinds of facts are to be recorded about different members of the hierarchy. If these types were not represented separately, we would have a single large relation with may occurrences of the special null value which means "value inapplicable."

Generalization inclusion relationships are a fact of life in most commercial database applications. However, most commercially available database management systems offer little or no generalized support for maintaining these relationships. Therefore, the DBA usually deals with them in an ad hoc manner. One approach is to define separate relation schemes corresponding to the entity types in the generalization hierarchy, but to rely on application program logic to maintain the inclusion relationships. For example, in place of a generalized subtype integrity constraint mechanism, application program logic would enforce rules such as the following: for any tuple in the salesman relation, a corresponding tuple must exist in the employee relation with the same entity identifier value.

Another approach is to place together in one relation scheme all of the attributes which are conceptually associated with entity types at different levels of the generalization hierarchy. In effect, this amounts to storing the data as a materialized outer join of the relations representing the various entity types in the hierarchy. For example, the outer natural join of the conceptual employee and salesman relations would be stored as a base relation in place of separate employee and salesman relations. Thus, the database design approach can create the potential for missing and inapplicable database values in base relations.

In addition to the previous comments, I would like to pose the following questions concerning missing and inapplicable database values. Can they, in all cases, be eliminated from base relations by placing attributes at the proper level in a generalization hierarchy? If not, can the remaining cases be eliminated by other means?

Frank D. Linn
Lockheed Missiles and Space Company, Inc.
1111 Lockheed Way, Sunnyvale CA 94086
(408) 742-9880 O/19-71 B/524

REFERENCES

1. Codd, E. F., "Missing Information (Applicable and Inapplicable) in Relational Databases", SIGMOD RECORD, Vol 15, No 4, Dec 1986

2. Codd, E. F., "Extending the Database Relational Model to Capture More Meaning", ACM TODS, Vol 4, No 4, Dec 1979

3. Smith, J. M., and Smith, D. C. P., "Database Abstractions: Aggregation and Generalization", ACM TODS, Vol 2, No2, June 1977