CODD'S 12 RULES

Mark E. Donaldson

The relational data model was first developed by Dr. E.F. Codd, an IBM researcher, in 1970. In 1985, Dr. Codd published a list of 12 rules that concisely defined an ideal relational database. These rules have been used as a guideline for the design of all relational database systems since then.

I use the term "guideline" because, to date, no commercial relational database system fully conforms to all 12 rules. They do represent the relational ideal, though.

For a few years, scorecards were being kept that rated each commercial product on how well they conformed to Codd's rules. Today, the rules are not talked about as much, but they still remain a goal for relational database design.

Following is Codd's 12 rules. His original name for each rule is listed with a simplified description. I have included a note where certain rules are problematic to implement. Don't worry if some of these items are confusing to you.

RULE 1: THE INFORMATION RULE
All data should be presented to the user in table form.

RULE 2: GUARANTEED ACCESS RULE
All data should be accessible without ambiguity. This can be accomplished through a combination of the table name, primary key, and column name.

RULE 3: SYSTEMATIC TREATMENT OF NULL VALUES
A field should be allowed to remain empty. This involves the support of a null value which is distinct from an empty string or a number with a value of zero. Of course, this can't apply to primary keys. Also, most database implementations support the concept of a nun-null field constraint that prevents null values in a specific table column.

RULE 4: DYNAMIC ON-LINE CATALOG BASED ON THE RELATIONAL MODEL
A relational database must provide access to its structure through the same tools that are used to access the data. This is usually accomplished by storing the structure definition within special system tables.

RULE 5: COMPREHENSIVE DATA SUBLANGUAGE RULE
The database must support at least one clearly defined language that includes functionality for data definition, data manipulation, data integrity, and database transaction control. All commercial relational databases use forms of the standard SQL (Structured Query Language) as their supported comprehensive language.

RULE 6: VIEW UPDATING RULE
Data can be presented to the user in different logical combinations, called views. Each view should support the same full range of data manipulation that direct access to a table has

available. In practice, providing update and delete access to logical views is difficult and is not fully supported by any current database.

RULE 7: HIGH-LEVEL INSERT, UPDATE, AND DELETE
Data can be retrieved from a relational database in sets constructed of data from multiple rows and/or multiple tables. This rule states that insert, update, and delete operations should be supported for any retrievable set rather than just for a single row in a single table.

RULE 8: PHYSICAL DATA INDEPENDENCE
The user is isolated from the physical method of storing and retrieving information from the database. Changes can be made to the underlying architecture (hardware, disk storage methods) without affecting how the user accesses it.

RULE 9: LOGICAL DATA INDEPENDENCE
How a user views data should not change when the logical structure (tables structure) of the database changes. This rule is particularly difficult to satisfy. Most databases rely on strong ties between the user view of the data and the actual structure of the underlying tables.

RULE 10: INTEGRITY INDEPENDENCE
The database language (like SQL) should support constraints on user input that maintain database integrity. This rule is not fully implemented by most major vendors. At a minimum, all databases do
Preserve two constraints through SQL.

- No component of a primary key can have a null value. (see rule 3)
- If a foreign key is defined in one table, any value in it must exist as a primary key in another table.

RULE 11: DISTRIBUTION INDEPENDENCE
A user should be totally unaware of whether or not the database is distributed (whether parts of the database exist in multiple locations). This is difficult to implement for a variety of reasons that we will
spend time on in future newsletters when we discuss distributed databases.

RULE 12: NONSUBVERSION RULE
There should be no way to modify the database structure other than through the multiple row database language (like SQL). Most databases today support administrative tools that allow some direct manipulation of the data structure.