Relational Database Management Systems

Chapter Eight: Attribute Data and Relational Database Management Systems
Attribute database tables
Creating database tables
Working directly with external database management systems
Managing database tables
Tables and views
Grouping data
Relational databases
Adding the spatial dimension
RDBMS and data integration
Written Exercise 8: Integrating data using an RDBMS

Common RDBMSs
Microsoft Access
Oracle
MySQL
System to manage databases based on relational model.
Basics
Data are stored in tables.
In any field with repeated data, we create a new table and substitute a lookup code in the first table and define a relationship between the two tables.

Example

Name	Major	Year	Transport	Food	City	State	Zip
Ali	sociology	sophomore	bike	vegan	Oakland	CA	94613
Beth	environmental studies	junior	bike	omni	Oakland	CA	94602
Carlos	sociology	fresh	car	pesc	Oakland	CA	94604
Deni	biology	senior	foot	omni	San Rafael	CA	96567
Edith	biology	junior	foot	veg	Pullman	WA	99231
Farrel	sociology	fresh	foot	veg	Eugene	OR	98992
Gerri	biology	junior	car	veg	Prescott	AZ	89432

Codd’s 12 Rules

Rule 0: The system must qualify as *relational*, as a *database*, and as a *management system*.

For a system to qualify as a relational database management system (RDBMS), that system must use its *relational* facilities (exclusively) to *manage* the *database*.

Rule 1: The *information rule*:

All information in the database is to be represented in one and only one way, namely by values in column positions within rows of tables.

Rule 2: The *guaranteed access rule*:

All data must be accessible. This rule is essentially a restatement of the fundamental requirement for primary keys. It says that every individual scalar value in the database must be logically addressable by specifying the name of the containing table, the name of the containing column and the primary key value of the containing row.

Rule 3: *Systematic treatment of null values*:

> The DBMS must allow each field to remain null (or empty). Specifically, it must support a representation of "missing information and inapplicable information" that is systematic, distinct from all regular values (for example, "distinct from zero or any other number", in the case of numeric values), and independent of data type. It is also implied that such representations must be manipulated by the DBMS in a systematic way.

Rule 4: *Active online catalog based on the relational model*:

> The system must support an online, inline, relational catalog that is accessible to authorized users by means of their regular query language. That is, users must be able to access the database's structure (catalog) using the same query language that they use to access the database's data.

Rule 5: The *comprehensive data sublanguage rule*:

> The system must support at least one relational language that
>
> 1. Has a linear syntax
> 2. Can be used both interactively and within application programs,
> 3. Supports data definition operations (including view definitions), data manipulation operations (update as well as retrieval), security and integrity constraints, and transaction management operations (begin, commit, and rollback).

Rule 6: The *view updating rule*:

> All views that are theoretically updatable must be updatable by the system.

Rule 7: *High-level insert, update, and delete*:

> The system must support set-at-a-time *insert*, *update*, and *delete* operators. This means that 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*:

> Changes to the physical level (how the data is stored, whether in arrays or linked lists etc.) must not require a change to an application based on the structure.

Rule 9: *Logical data independence*:

Changes to the logical level (tables, columns, rows, and so on) must not require a change to an application based on the structure. Logical data independence is more difficult to achieve than physical data independence.

Rule 10: *Integrity independence*:

Integrity constraints must be specified separately from application programs and stored in the catalog. It must be possible to change such constraints as and when appropriate without unnecessarily affecting existing applications.

Rule 11: *Distribution independence*:

The distribution of portions of the database to various locations should be invisible to users of the database. Existing applications should continue to operate successfully :

1. when a distributed version of the DBMS is first introduced; and
2. when existing distributed data are redistributed around the system.

Rule 12: The *nonsubversion rule*:

If the system provides a low-level (record-at-a-time) interface, then that interface cannot be used to subvert the system, for example, bypassing a relational security or integrity constraint.