Codd’s 12 Rules

E.F Codd was a Computer Scientist who invented **Relational model** for Database management. Based on relational model, **Relation database** was created. Codd proposed 13 rules popularly known as **Codd's 12 rules** to test DBMS's concept against his relational model. Codd's rule actually define what quality a DBMS requires in order to become a Relational Database Management System(RDBMS). The Codd's 12 rules are as follows

Rule 0: The *Foundation rule*:

A relational database management system must manage its stored data using only its relational capabilities. 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 a relational database (including table and column names) is represented in only one way, namely as a value in a table.

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.