The 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 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.

Database Security

Database security has many different layers, but the key aspects are:

Authentication

User authentication is to make sure that the person accessing the database is who he claims to be. Authentication can be done at the operating system level or even the database level itself. Many authentication systems such as retina scanners or bio-metrics are used to make sure unauthorized people cannot access the database.

Authorization

Authorization is a privilege provided by the Database Administer. Users of the database can only view the contents they are authorized to view. The rest of the database is out of bounds to them.

The different permissions for authorizations available are:

- **Primary Permission -** This is granted to users publicly and directly.
- **Secondary Permission -** This is granted to groups and automatically awarded to a user if he is a member of the group.
- **Public Permission -** This is publicly granted to all the users.
- **Context sensitive permission -** This is related to sensitive content and only granted to a select users.

The categories of authorization that can be given to users are:

- **System Administrator -** This is the highest administrative authorization for a user. Users with this authorization can also execute some database administrator commands such as restore or upgrade a database.
- **System Control -** This is the highest control authorization for a user. This allows maintenance operations on the database but not direct access to data.
- **System Maintenance -** This is the lower level of system control authority. It also allows users to maintain the database but within a database manager instance.
- **System Monitor -** Using this authority, the user can monitor the database and take snapshots of it.

Database Integrity

Data integrity in the database is the correctness, consistency and completeness of data. Data integrity is enforced using the following three integrity constraints:

- **Entity Integrity -** This is related to the concept of primary keys. All tables should have their own primary keys which should uniquely identify a row and not be NULL.

- **Referential Integrity -** This is related to the concept of foreign keys. A foreign key is a key of a relation that is referred in another relation.
- **Domain Integrity -** This means that there should be a defined domain for all the columns in a database.