Introduction of Codd's Rule for Relational DBMS
By

Prof. Vinod S. Ramteke

Assistant Professor
Department of Computer Science
Janata Mahavidyalara , Chandrapur(M.S.)

Introduction

- E.F Codd was a Computer Scientist who invented the **Relational model** for Database management. Based on relational model, the **Relational 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 actualy define what quality a DBMS requires in order to become a Relational Database Management System (RDBMS).

Codd's Rules Zero

- *This rule states that for a system to qualify as an **RDBMS**, it must be able to manage database entirely through the relational capabilities.*

Rule 1: Information rule

- All information(including metadata) is to be represented as stored data in cells of tables. The rows and columns have to be strictly unordered.

Rule 2: Guaranteed Access

- *Each unique piece of data(atomic value) should be accessible by : **Table Name + Primary Key(Row) + Attribute(column)**.*

Rule 3: Systematic treatment of NULL

- *Null has several meanings, it can mean missing data, not applicable or no value. It should be handled consistently. Also, Primary key must not be null, ever. Expression on NULL must give null.*

Rule 4: Active Online Catalog

- *Database dictionary(catalog) is the structure description of the complete **Database** and it must be stored online. The Catalog must be governed by same rules as rest of the database. The same query language should be used on catalog as used to query database.*

Rule 5: Powerful and Well-Structured Language

- *One well structured language must be there to provide all manners of access to the data stored in the database. Example: **SQL**, etc. If the database allows access to the data without the use of this language, then that is a violation.*

Rule 6: View Updation Rule

- *All the view that are theoretically updatable should be updatable by the system as well.*

Rule 7: Relational Level Operation

- *There must be Insert, Delete, Update operations at each level of relations. Set operation like Union, Intersection and minus should also be supported.*

Rule 8: Physical Data Independence

- *The physical storage of data should not matter to the system. If say, some file supporting table is renamed or moved from one disk to another, it should not effect the application.*

Rule 9: Logical Data Independence

- *If there is change in the logical structure(table structures) of the database the user view of data should not change. Say, if a table is split into two tables, a new view should give result as the join of the two tables. This rule is most difficult to satisfy.*

Rule 10: Integrity Independence

- *The database should be able to enforce its own integrity rather than using other programs. Key and Check constraints, trigger etc, should be stored in Data Dictionary. This also make **RDBMS** independent of front-end.*

Rule 11: Distribution Independence

- *A database should work properly regardless of its distribution across a network. Even if a database is geographically distributed, with data stored in pieces, the end user should get an impression that it is stored at the same place. This lays the foundation of **distributed database**.*

Rule 12: Nonsubversion Rule

- *If low level access is allowed to a system it should not be able to subvert or bypass integrity rules to change the data. This can be achieved by some sort of looking or encryption.*

Ok..............

THANKS ....

TO

ALL.................