Jump2Learn
PUBLICATION

www.jump2learn.com

CONCEPTS of RELATIONAL DATABASE MANAGEMENT SYSTEM

Jump2Learn - The Online Learning Place

Dr. RajeshKumar R. Savaliya | Ms. Sonal B. Shah | Mr. Vaibhav D. Desai

Unit-1

Introduction of Relational Model

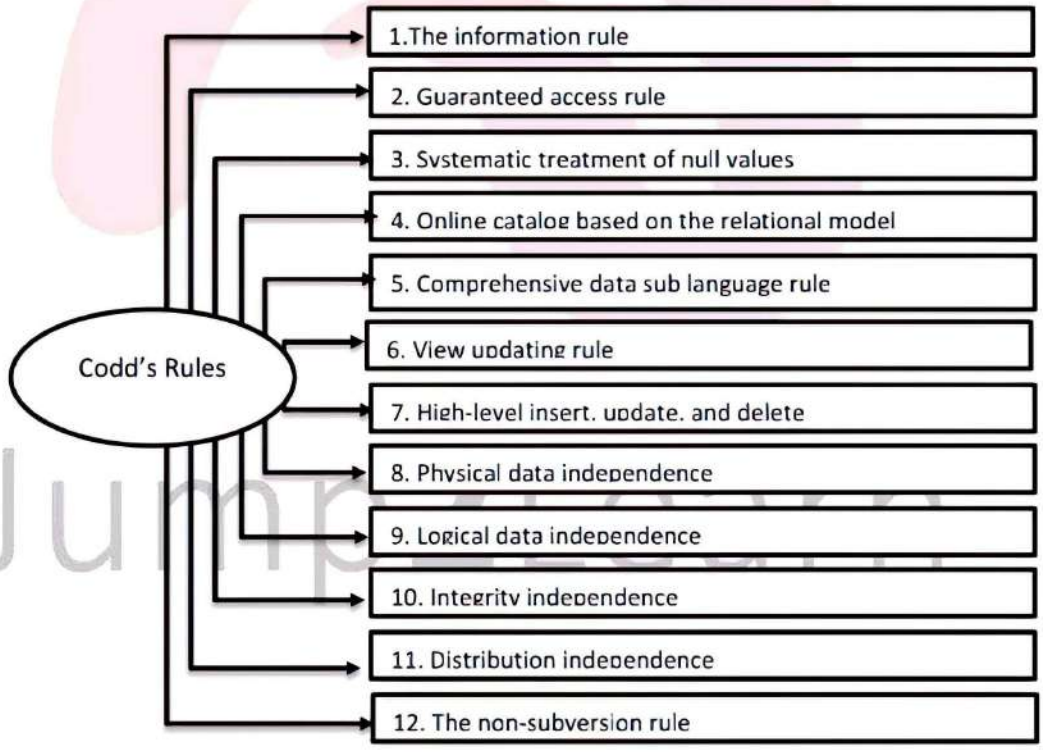
1.1Codd's Rules

Dr Edgar F. Codd was one popular computer scientist who designed the Relational model for Database management system. Dr E.F. Codd was proposed thirteen rules (zero to twelve) for Database management system and if a Database Management System satisfies more than 7 to 8 codd rules that can be known as a Relational Database Management System. List of these 13 Codd's laws are as follow.

Zero Rule (The Foundation Rule)

The database system must be suitable as relational database model. A database system to become as a relational database management system (RDBMS) that must be uses completely relational abilities to manage data and database.

The remain other twelve rules derive from this base rule and List of these 12 Codd's laws are as follow.



1) The information rule

The database holds various data and these data must be stored in a table in the form of rows and columns. For example: we can store the personal information (Name, Address, City, Mobile_No, etc.) of student in STUDENT table in the form of rows and columns.

2) Guaranteed access rule

The database holds various data and these data must be accessible with the help of table name, column name and primary key.

3) Systematic treatment of null values

The database must allow each field to remain empty (or null). It supports the null value and null value is different from number with value zero as well as empty string.

4) Online catalog(structure) based on the relational model

The database must have provision of online and which should be accessible to authorized users using query language (sql). A relational database must be providing the access to database's structure (catalog) with the help of same query language that are used to access the database's data.

5) Comprehensive data sub language rule

The database must support at least one database language that consist of database functionality such as data definition, data manipulation, data integrity, data query and database transaction management operation (commit, rollback, Savepoint).

6) View updating rule

Data can be presented in different logical combinations called view. All the view should support data manipulation (data insert, data update and data delete) in relational database.

7) High-level insert, update, and delete

The relational database system must keep an eye on high-level relational operations such as insert, update, and delete on a multiple row across a multiple table.

8) Physical data independence

Physical data independence is used to support for changing any hardware without affecting to database structure. All data stored in a relational database must be physically independent and it is also not affected when changing any hardware in system.

9) Logical data independence

Logical data independence is similar to physical data independence. If any changes happened to the table structures (logical level), it must not affect the user's viewed data or application data.

10) Integrity independence

The relational database must be supports integrity independence when we are inserting data into table's cells using the SQL query language. A database must be independent than the application that uses it. All integrity constraints can be individually modified without the affecting the application.

11) Distribution independence

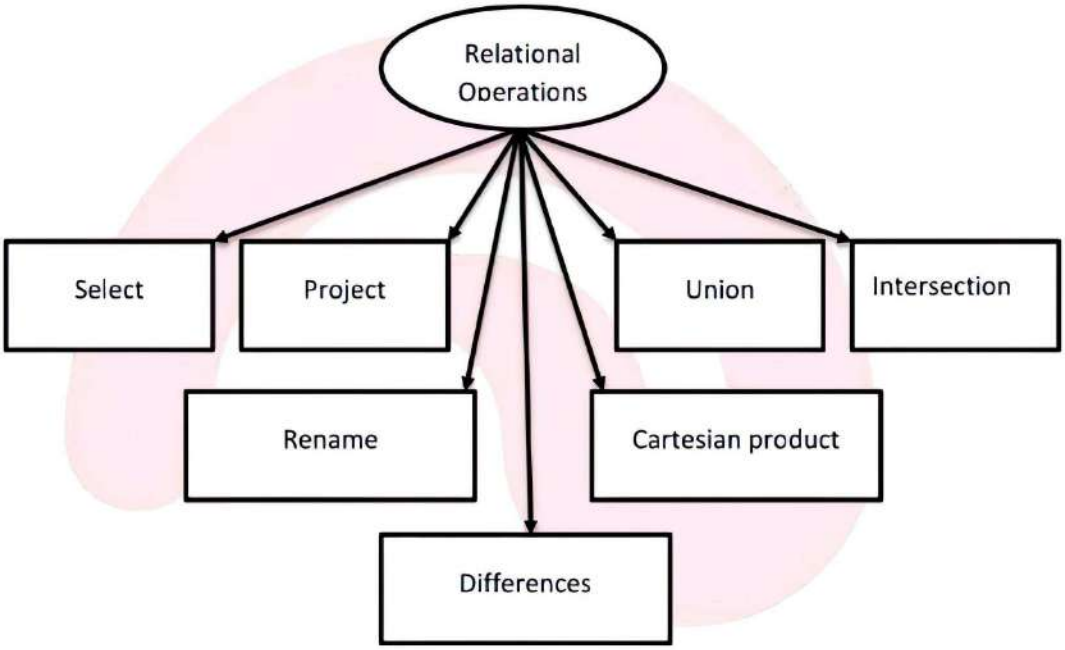
If portion of database exist in multiple location and then it must be possible to perform relational operations (insert, update, delete, etc.) upon it that is known as Distribution independence.

12) The non-subversion rule

The database structure (catalog) should not be modifying without use of database language such as SQL.

1.2 Relational operations Algebra

Relational algebra is the procedural query language which takes relation as input and produce relation as output. It provides a step by step procedure to get the result of the query. There are two types of operator can be used such unary or binary. The primary operations that we can perform using relational algebra are as follow:



1) Select Operation

The select operation selects number of rows based on given conditions. The select operation is denoted by **sigma (σ)**. Notation (syntax) of select operation is denoted by: **σ p(r)**.

Where:

σ is used for selection statement (select).
r is used for name of relation (table).
p is used for logical formulas or conditions (using logical operators such as AND , OR and NOT as well as relational operators such as =, ≠, ≥, <, >, ≤.).

Suppose we have one Employee relation(table) with data that are as follows:

Emp_No	Emp_Name	Emp_Age	Emp_City	Mobile_No
101	Rajesh	36	Surat	9510103123
102	Vaibhav	38	Surat	9510104123
103	Sonal	37	Vadodara	9510105123
104	Swara	25	Bharuch	9510106123
105	Darshan	30	Vadodara	9510107123

Example 1: -
Now we want to fetch data from Employee table with whose age more than 36.

σ age >= 36 (Employee)

This select operation will select the rows from Employee table, for which age will be greater than or equal 36.

Output: -

Emp_No	Emp_Name	Emp_Age	Emp_City	Mobile_No
101	Rajesh	36	Surat	9510103123
102	Vaibhav	38	Surat	9510104123
103	Sonal	37	Vadodara	9510105123

we can also use the logical operators to specify more than one conditions as follow.

Example 2:

σ age >= 36 and city = 'Surat' (Employee)

This select operation will return rows from Employee table with information of Employee whose city name is Surat and age more than 36.

Output: -

Emp_No	Emp_Name	Emp_Age	Emp_City	Mobile_No
101	Rajesh	36	Surat	9510103123
102	Vaibhav	38	Surat	9510104123

2) Project Operation

Project operation is used to display only selected attributes as a result and rest of the attributes are not included in result. Project operation is denoted by ∏. Notation (syntax) of Project operation is denoted by: **∏ A1, A2,... An. (r)**

Where:

∏ is used for Project operation.
r is used for name of relation (table).
A1, A2, An is used as an attribute name of relation (table) r.

Example 1: -

Now we want to fetch only two attribute Name and City from Employee table.

∏ Emp_Name, Emp_City (Employee)

This select operation will select only two attributes with all rows from Employee table.

Output: -

Emp_Name	Emp_City
Rajesh	Surat
Vaibhav	Surat
Sonal	Vadodara
Swara	Bharuch
Darshan	Vadodara

3) Union Operation

The union operation contains all the rows that are either in R or S or both in R & S. Union operation removes the duplicate rows from R and S. It is denoted by **U**. Notation (syntax) of union operation is denoted by: **R U S**

A union operation must hold the condition such as R and S must have the attribute of the same type and Duplicate rows are removed automatically. We have two relations as follow.

Depositor table:

Name	Account_No
Rajesh	1
Vaibhav	2
Sonal	3
Swara	4
Darshan	5

Borrower table:

Name	Loan_No
Rajesh	10
Minesh	20
Foram	30

Example 1: -

∏ Name (Depositor) ∪ ∏ Name (Borrower)

Output: -

Name
Rajesh
Vaibhav
Sonal
Swara
Darshan
Minesh
Foram

4) Intersection Operation

The intersection operation contains all the rows that are in both R or S. Intersection operation only retrieve the common rows from R and S. It is denoted by ∩. Notation (syntax) of intersection operation is denoted by: **R ∩ S**. We have two relations as follow.

Depositor table:

Name	Account_No
Rajesh	1
Vaibhav	2
Sonal	3
Swara	4
Darshan	5

Borrower table:

Name	Loan_No
Rajesh	10
Minesh	20
Foram	30

Example 1: -

∏ Name (Depositor) ∩ ∏ Name (Borrower)

Output: -

Name
Rajesh

5) Differences Operation

The set difference operation contains all tuples that are in R but not in S. It is denoted by intersection minus **(-)**. Notation (syntax) of differences operation is denoted by: R – S. We have two relations as follow.

Depositor table:

Name	Account_No
Rajesh	1
Vaibhav	2
Sonal	3
Swara	4
Darshan	5

Borrower table:

Name	Loan_No
Rajesh	10
Minesh	20
Foram	30

Example 1: -

∏ Name (Depositor) - ∏ Name (Borrower)

Output: -

Name
Vaibhav
Sonal
Swara
Darshan

6) Rename Operation

The rename operation is used to rename the output relation. It is denoted by ρ. Notation (syntax) of rename operation is denoted by: ρ(Name_of_New_Relation, Name_of_Old_Relation)

Example 1: -

We can use the rename operator to rename Employee relation to Emp.

ρ (Emp, Employee)

Output: -

Emp Relation(table) after rename operation.

Emp_No	Emp_Name	Emp_Age	Emp_City	Mobile_No
101	Rajesh	36	Surat	9510103123
102	Vaibhav	38	Surat	9510104123
103	Sonal	37	Vadodara	9510105123
104	Swara	25	Bharuch	9510106123
105	Darshan	30	Vadodara	9510107123

7) Cartesian product Operation

The Cartesian product operation is used to combine each row in one table with each row in the other table. It is also known as a cross product operation.
It is denoted by **X**. Notation (syntax) of Cartesian product operation is denoted by: **E X D**

We can use the Cartesian Product Operation with Employee Relation Dept_details relation.

Employee relation

Emp_No	Emp_Name	Emp_Age
101	Rajesh	36
102	Vaibhav	38
103	Sonal	37

Dept_Details relation

Dept_Name	Emp_Salary
BCA	51000
BBA	52000
BCOM	53000

Example 1: -

Employee X Dept_Details

Output: -

Emp_No	Emp_Name	Emp_Age	Dept_Name	Emp_Salary
101	Rajesh	36	BCA	51000
101	Rajesh	36	BBA	52000
101	Rajesh	36	BCOM	53000
102	Vaibhav	38	BCA	51000
102	Vaibhav	38	BBA	52000
102	Vaibhav	38	BCOM	53000
103	Sonal	37	BCA	51000
103	Sonal	37	BBA	52000
103	Sonal	37	BCOM	53000

1.3 Transaction control language

The series of operation perform on oracle table data is called oracle transaction. Transaction Control Language(TCL) commands are used to handle the transactions in the database. There are main three type of transactions.

1) COMMIT
2) SAVEPOINT
3) ROLLBACK

1) COMMIT

The commit ends the current transaction and save the changes made during the transaction. The commit is also used to save modification mad during the transaction.

Syntax :-

Commit;

Example :

Commit;

2) SAVEPOINT

Savepoint is used to create the point within oracle transactions and also save the all current transactions are done. We can rollback latter since specified savepoint.

Syntax: -

Savepoint <save_pointname>;

Example:

```
insert into Employee values(106,'sagar',41,'surat',9510108123);
savepoint sp;
insert into Employee values (107,'parth',42,'vapi', 9510109123);
rollback to savepoint sp;
```

Output: (select * from Employee;**)**

Emp_No	Emp_Name	Emp_Age	Emp_City	Mobile_No
101	Rajesh	36	Surat	9510103123
102	Vaibhav	38	Surat	9510104123
103	Sonal	37	Vadodara	9510105123
104	Swara	25	Bharuch	9510106123
105	Darshan	30	Vadodara	9510107123
106	Sagar	41	Surat	9510108123

3) ROLLBACK

Rollback does exactly the opposite of commit. It ends the transaction but undoes(undo) any changes made during the transaction.

Syntax: -

Rollback [to [save point] <save point name>];

Example 1:

Rollback;

Example 2:

Rollback to savepoint sp;

1.4 Data Control language:

Oracle provides security features for data store in table Data Control Language(DCL) commands are used to handle the database security when we working on single database with multiple user environment. There are two types of DCL commands or privileges are as follow.

1) GRANT
2) REVOKE

1) GRANT

GRANT command used to give the access permission or privileges on the database objects to the users based on object privileges given to particular user.

Syntax: -

GRANT <Object Privileges>
ON <Object Name>
TO < User Name>
[WITH GRANT OPTION]

Object Privileges

Object privilege name is the access right or privilege granted to the user. List of the privileges such as SELECT, INSERT, UPDATE, DELETE, ALTER, INDEX.

Object Name

Object name is the name of database object like TABLE, VIEW, STORED PROCEDURE and SEQUENCE.

User Name

User name is the name of the user to whom an access right or privilege is being granted.

WITH GRANT OPTION

The "WITH GRANT OPTION" allows a user to grant access rights or privilege to other users.

Example:

GRANT select, insert ON Employee TO system;

2) REVOKE

The REVOKE command is used to take back access rights or privileges from the user that is given by the GRANT command on the database objects.

Syntax: -

REVOKE <Object Privileges>
ON <Object Name>
FROM < User Name>

Example:

REVOKE select, insert ON Employee FROM system;

In this example, REVOKE command is used to revoke the object privileges those are granted by the GRANT command such as the SELECT and INSERT privileges on Employee table from system. When you REVOKE SELECT and INSERT privilege on a table from a user (system), the user will not be able to SELECT and INSERT data from that table Employee.

The REVOKE command cannot be used to revoke the access right or privileges granted by the operating system.

****