UNIT III RELATIONAL DATABASE DESIGN

RELATIONAL MODEL

RDBMS is a collection of tables having unique names

Attributes and Domains

Some commonly used terms in RDB model are

- Table / relation.

They are collections of data items arranged in rows & columns. The table cannot have duplicate data or rows

- Tuple / record / row.

- It is a part of table that contain several records.
- Each record can be broken down into several small parts of data known as attributes
- Each record is a single row / entity represented as a set of related data.

Relation Schema:

A relation schema describes the structure of the relation with the name of the relation its attribute example: Name of table, or their names and type.

Relation Instance:

It refers to specific instance of relation i.e containing a specific set of rows.

Domain

For each attribute of relation, there is a set of permitted values called domain.

Example domain of Marks (83, 88, 98)

- Atomic

The domain is atomic if elements of the domain are considered to be indivisible units.

NULL attribute:

A null is a special symbol, independant of data types, which means either unknown or inapplicable. It does not mean zero or blank

Degree:

It is nothing but total number of columns present in the relational database.

Cardinality:

It is total number of tuples present in the relational database.

CODD's 12 RULES

Rule 0: for a database to be relational it must use its relational capabilities to manage the database.

Rule 1: Information Rule All info in RDBMS is represented logically only by storing in tables

Rule 2: Guarenteed Access: Each item of data in RDMS is guarenteed to be accessible by specifying that table, primary key and column name

Rule 3: Systematic treatment of null values: Null values are supported in RDBMS for incomplete or inapplicable info in systematic way which is independant of data type

Rule 4: Dynamic online catalog based on Relational Model: Database dictionary called as catalog. - is a structured description of complete database and it must be stored online -
This catalog must be governed by same rules as rest of the database

Rule 5: Comprehensive Data Sublanguage use: At least 1 well structured, well defined language must be there which can access all the data present in database.

Rule 6: View updating Rule: All views in DBS which are theoretically updatable must be updatable in practice by DBMS

Rule 7: relational level operation. The High level Insert, Update and Delete rule: There must be insert, delete and update operation at each level of relations

Rule 8: The physical Data Independance rule: If any changes are made in either storage representation or access methods then it should not affect the application.

Rule 9: The Logical Data Independance rule: If any changes are made in table structure then the logical view of the user should not get affected. FPR Rule. If a table is split internally the view creates must have the "ENTIRE" table.

Rule 10: The Integrity Independance rule: The integrity constraint must be defined by the RDMS stored in the system and it should not be enforced by the external application programs.

Rule 11: The Distribution Independance rule: An RDBMS must have distribution independance. That means even if database is scattered geographically, user should get a feel as if it is stored at one place.

Rule 12: The Non Sub version rule: If low level lang is allowed to access the system. then that low level lang must not be able to bypass / subvert the integrity rules which are expressed at high level lang.

RELATIONAL INTEGRITY

[Refer Unit I : Keys]

[ONSTRAINT :

Constraint mean some rule that have been set on DB
There are 3 main type of constraints.

i] Domain Constraint:

- Domain constraint defines the domain or set of values for an attribute
- The datatype of a domain include string, char, int, time, date, currency etc. The value of attribute must be available in corresponding domain.

ii] key or Null Constraint :

- keys are used to identify particular record from the tables. Primary key is usually used to identify record uniquely
- Hence key constraint can be stated as.
  - All values of pk must be unique
  - The pk must not be null.

iii] Integrity Constraint:

- Integrity constraint are the rules that are to be applied on database columns to ensure validity of data.
- example: - EmpID must consist of two digits
  - EmpID must begin with a letter.

a) Entity Integrity Constraint:

- The value of attribute of primary key can not be null in a relation.
- Null represent incomplete or exceptional data.
- The pk helps uniquely identify every row in the table
- Thus for a record to be accessible pk != Null.

b) Referential Integrity Constraint:

- In Relationship data is linked between two or more tables
- This is achieved by using foreign keys referencing a primary key from parent table. Because we need to ensure that data on both sides of relationship remain intact
- The referential integrity rule states that "whenever a foreign key value is used it must reference a valid, existing primary key value in parent table"

ENTERPRICE CONSTRAINTS:

Also called as semantic constraints

They are constrains enforced by user or database administrator on multiple tables

DATA BASE DESIGN

Features of good Database design.

There are two goals of relational database design

i] To generate a set of relation schema that allows us to store information without unnecessary redundancy

ii] To allow us to retrieve information easily.

Data redundancy and update Anomalies

Data redundancy is a condition created in database in which same piece of data is held at two places.

- Redundancy is at the root of several problems associated with relational schema.

i] Redundant Storage : Some information is stored repeatedly

ii] Update Anomalies : If one copy of such repeated data is updated then inconsistency is created unless all other copies are similarly updated.

iii] Insertion Anomalies : Due to insertion of new record repeatedly same information get added to the relational schema.

iv] Deletion anomalies : Due to ~~hidden~~ deletion of particular records some other important information associated with the deleted record get deleted and thus we may ~~be~~ lose some other important information ~~from~~ the schema.

NORMALIZATION

Normalization is a process of reorganizing data in a database so that it meets two basic requirements:

1> There is no redundancy of data

2> Data dependancy are logical

Need for Normalization.

- It eliminates redundant data.
- It reduces chances of data error.
- It allows database to take up less disk space.
- It also helps increase the performance
- It ~~depends~~ improves data integrity and consisteney.

Atomic domain & 1ST NORMAL FORM

By atomic value we mean that each value stored is indivisible

1NF :

It should have atomic valued attribute.
Values stored in ~~domain~~ column should be of same domain.
All the column should have unique names
The order in which data is stored doesn't matter.

2NF :

The value should be in first Normal form
It should not have partial functional dependancy.

3NF :

It should be in ~~this~~ second Normal form
It doesn't have transitive dependancy
i.e $x \rightarrow y$, $y \rightarrow z$ $\therefore x \rightarrow z$ transitive
thus x - super key / y - prime attribute

BCNF : Boyce-Codd

It should be in third Normal form
If $x \rightarrow y$ then x should be super key i.e
if y is prime attribute then x can not be non prime

4NF :

It should be in BCNF
No Multivalued dependancy

5NF :

It should be in 4NF
should be lossless decomposition.

Decomposition using Functional Dependancies

A functional dependancy $A \rightarrow B$ in a relation holds if two tuples having same values of attribute A also having the same value for attribute B. It is denoted by $A \rightarrow B$ where A is called determinant and B is called dependant.

Trivial FD : The functional dependancy $A \rightarrow B$ is trivial if B is a subset of A i.e $A \rightarrow A$ or $AB \rightarrow A$ ($lhs \cap Rhs \; != \phi$ then)

Non Trivial FD : The functional dependancy $A \rightarrow B$ is non trivial if B is not a subset of A $lhs \cap Rhs = \phi$ then non trivial.

Algorithm for Decomposition.

Decomposition is the process of breaking down one table into multiple tables.

Problems related to decomposition:

1) Some query becomes more expensive
2) Given instance of the decomposed relations, we may not be able to reconstruct the corresponding instance of the original relation.
3) Checking some dependancies may require joining the instance of the ~~development~~ decomposed relations.
4) There may be loss of information during decomposition.

Properties associated with decomposition:

There are two properties associated with decomposition

i) Union : attribute of R1 & R2 must be equal to attribute of R. Each attribute of R must be either in R1 or in R2

$$Att(R_1) \cup Att(R_2) = Att(R)$$

ii) Intersection : attribute of R1 & R2 must not be NULL

$$Att(R_1) \cap Att(R_2) \neq \phi$$

iii) Common : must be a key for atleast one relation (R1 or R2)

$$Att(R_1) \cap Att(R_2) \rightarrow Att(R_1) \quad \text{or}$$
$$Att(R_1) \cap Att(R_2) \rightarrow Att(R_2)$$

DEPENDANCY PRESERVING DECOMPOSITION

example

Let R(ABCD) with FD {A→B, B→C, C→D, D→B}

R is decomposed into $R_1(AB)$, $R_2(BC)$, $R_3(BD)$

$A^+ = B,C,D \mid B^+ = C,D \mid C^+ = D,B \mid D^+ = BC$ – closures

$R_1(AB)$

A→A	– trivial	– x
A→B	– B in A^+	✓
B→A	– A not in B^+	– x
B→B	– trivial	– x

$R_2(BC)$

B→B	– trivial	– x
B→C	– C in B^+	✓
C→B	– B in C^+	✓
C→C	– trivial	– x

$R_3(BD)$

B→B	– trivial	– x
B→D	– D in B^+	✓
D→B	– B in D^+	✓
D→D	– trivial	– x

∴ All possible FD's are. – ①.

A→B
B→C
C→B
B→D
D→B

Lets check if FD is preserved.

A→B is in – ①. B→C is in – ①

E→D not in – ① but if we take E^+ in 1 $\{E^+ = BD\}$ thus preserved

D→B is in – ①

∴ All dependancy are preserved. as.

① No attribute loss $R_1 \cup R_2 \cup R_3 = R$,

② Dependancy of R in R_1, R_2, R_3.

Lossless vs Lossy Decomposition:

If the table contains values/tuples which were not in original table after implementing a join then it is lossy decomposition.

(extra tuples are called spurious tuples).

If there are no spurious tuples then there is lossless decomposition.

∴ the common attribute should be superkey or candidate key.