Unit-4

The Relational Data Model and Relational Database Constraints:

⊛ Relational Model Concepts:

Relational model represents the database as a collection of relations. A relation is nothing but a table of values. Every row in the table represents a collection of related data values. The table name and column names are helpful to interpret the meaning of values in each row. The data are represented as a set of relations.

Domain → A domain is a set of acceptable values that a column is allowed to contain. This is based on various properties and the data type for the column. It is the original set of atomic values used to model data. Atomic value mean that each value in the domain is indivisible as far as the relational model is concerned.

For example:- The domain of Marital Status has a set of possibilities: Married, Single, Divorced.

Attribute → Attributes are the properties that define a relation.
For example:- ROLL_NO, NAME. etc.

Tuple → Each row in the relation is known as tuple. It contains the data records.

Relation → A relation in relational data model represents the respective attributes and the correlation among them.

Student (Relational Model).

Stu_id	Stu_name	Stu_address	Dept_id
12	Ramesh	Butwal	5
13	Hari	Kathmandu	3
14	Mahesh	Pokhara	1
15	Prem	Nepaljung	2
16	Sangita	Mahendranagar	4

Attributes (columns)

Record or Tuple (Row)

Relation (Table)

Primary key attribute.

⊛. Characteristics of Relations:

i) Each relation in a database must have a distinct or unique name which would seperate it from the other relations in a database.

ii) A relation must not have two attributes with same name. Each attribute must have a distinct name.

iii) Duplicate tuples must not be present in relation.

iv) Each tuple must have exactly one data value for an attribute.

v) Tuples in a relation do not have to follow a significant order as the relation is not order-sensitive.

vi) Similarly, the attributes of a relation also do not have to follow a significant order.

⊛. Relational Model Notation:-

We will use following notation in our presentation:

i) A relation schema R of degree n is denoted by $R(A_1, A_2, \ldots, A_n)$.

ii) The uppercase letters Q, R, S denote relation names.

iii) The lowercase letters q, r, s denote relation states.

iv) The letters t, u, v denote tuples.

v) The name of relation schema such as STUDENT indicates current set of tuples in that relation whereas STUDENT(Name, Roll, ...) refers only to relation schema.

vi) An attribute A can be qualified with the relation name R to which it belongs by using the dot notation R.A — for example: STUDENT.Name, STUDENT.Age etc.

vii) A n-tuple t in a relation $r(R)$ is denoted by $t = \langle v_1, v_2, \ldots, v_n \rangle$ where v_i is the value corresponding to attribute A_i.

→ Both $t[A_i]$ and $t.A_i$ (and sometimes $t[i]$ refer to the value v_i in t for attribute A_i.

→ Both $t[A_u, A_w, \ldots, A_z]$ and $t.(A_u, A_w, \ldots, A_z)$, where $A_u, A_w, \ldots, A_z$ is a list of attributes for R, refer to the subtuple of values $\langle v_u, v_w, \ldots, v_z \rangle$ from t corresponding to the attributes specified in the list.

⊛. Categories of Constraints:

The types of constraints are as follows:-

i) **Domain Constraints** → Domain constraints specify that within each tuple, the value of each attribute A must be an atomic value from the domain dom(A). The data types associated with domains typically include standard numeric data types for integers (such as short integer, integer, and long integer) and real numbers (float and double-precision float). Characters, Booleans, fixed-length strings are also available.

ii) **Key Constraints** → These are called uniqueness constraints since it ensures that every tuple in the relation should be unique. A relation can have multiple keys or candidate keys out of which we choose one of the key as primary key. We do not have any restriction on choosing the primary key out of candidate keys, but it is suggested to go with the candidate key with less number of attributes. NULL values are not allowed in the primary key, hence Not NULL constraint is also a part of key constraint.

iii) **Referential Integrity Constraints** → The referential integrity constraints is specified between two relations or tables and used to maintain the consistency among the tuples in two relations. This constraint is compulsory through foreign key. When an attribute in the foreign key of relation R1 ~~is said~~ have the same domain(s) as the primary key of relation R2, then the foreign key of R1 is said to refrence or refer to the primary key of relation R2. The values of the foreign key in a tuple of relation R1 can either take the values of the primary key for some ~~tub~~ tuple in relation R2, or can take NULL values, but can't be empty.

⊛ Relational Database and relational database schemas:-

A relational database schema is a set of relation schemas $S = \{R_1, R_2, \ldots, R_m\}$ and a set of integrity constraints. A relational database state of S is a set of relation states $DB = \{r_1, r_2, \ldots, r_m\}$ such that each r_i is a state of R_i and such that the r_i relation states satisfy the integrity constraints.

In simple language, a relational database is an arrangement of relation states in such a manner that every relational database state fulfills the integrity constraints set on a relational database schema. When we refer to a relational database, we implicitly include both its schema and its current state. A database state that does not obey all the integrity constraints is ~~current state.~~ called not valid, and a state that satisfies all the constraints in the defined set of integrity constraints is called a valid state. In the context of relational database schema following points deserve a particular consideration:

i) A specific characteristic, that bears the same real-world concept may appear in more than one relationship with the same or a different name. For example; In Employees relation, Employee Id (EmpId) is represented in Vouchers as AuthBy and PrepBy.

ii) The specific real-world concept that appears more than once in a relationship should be represented by different names.

iii) The integrity constraints that are specified on database schema shall apply to every database state of that schema.

⊛ Entity Integrity, Referential Integrity and Foreign Key:-

Entity Integrity → The entity integrity constraint states that no primary key value can be NULL. This is because the primary key value is used to identify individual tuples in the relation. Having NULL values for the primary key implies that we cannot identify some tuples. For example if two or more tuples had NULL for their primary keys, then we may not be able to distinguish them if we try to refrence them from other relations.

Refrential Integrity Constraint → The refrential integrity constraint is specified between two relations and is used to maintain the consistency among the tuples in the two relations. Refrential integrity constraints typically arise from the relationships among the entities represented by the relation schemas. This constraint is compulsory through foreign key. The values of foreign key in a tuple of relation R.1 is ~~said~~ can either take the values of the primary key for some tuple in relation R2, or can take NULL values, but can't be empty.

Foreign Key → It is the field in the table that is primary key of another table. A foreign key may accept multiple NULL values. A foreign key cannot automatically create an index, clustered or non-clustered. However we can manually create an index on foreign key. We have multiple foreign keys in the table.

⊛ Concept of insert, delete and update operations:

Q. Explain the operations and constraints violations?

Ans: Updates and retrieval are the two categories of operations on the relational model. The basic types of updates are:-

i) Insert → We use this operation in order to add new tuple in a relation. It is capable of violating any of the four constraints.

ii) Delete → We perform this operation in order to remove or delete a tuple in a relation. Under, this operation we can remove a particular data record from a table. It can only violate the referential integrity ~~constant.~~ constraint.

iii) Modify / Update → This operation causes a change in the values of some characteristic of existing tuples or accounting data tables.

Note: Retrival constraints do not cause a violation of integrity constraints.

⊛ Concept of transactions:-

A transaction is an executing program that includes some database operations, such as reading from database, or applying insertions, deletions, or updates to the database. A database application program running against a relational database typically executes one or more transactions. At the end of the transaction, it must leave the database in a valid or consistent state that satisfies all the constraints specified on the database schema.

A single transaction may involve any number of retrival operations. and any number of update operations. A large number of commercial applications running against relational databases in online transaction processing (OLTP) systems are executing transactions at rates that reach several hundred per second.

⊛. Advantages of using Relational Model:

→ A relational data model is simpler than the hierarchical and network model.
→ This model is concerned with data rather than structure. So this can improve the performance of model.
→ This model is easy to use since tables consisting of rows and columns is simple to understand.
→ This model is data independence since structure of database can be changed without having to change any application.
→ It makes possible for high-level query language like SQL to avoid complex database navigation.

⊛. Disadvantages of using Relational model:

→ Few relational databases have limits on field lengths which can't be exceeded.
→ Relational databases can sometimes become complex as the amount of data grows.
→ Complex relational database systems may lead to isolated databases where the information can't be shared from one system to another.