

UNIT 4

Relational Model:

a. Relational Model Concepts:

Relational model can represent as a table with columns and rows. Each row is known as a tuple. Each table of the column is known as attribute.

A relational database has following major components:

1. Table
2. Record or Tuple
3. Field or Column name or Attribute
4. Domain
5. Instance
6. Schema
7. Keys

1. Table

A table is a collection of data represented in rows and columns. Each table has a name in database. For example, the following table “STUDENT” stores the information of students in database.

Table: STUDENT

id	Name	age	address
1	A	17	abc
2	B	16	def
3	C	20	pqr

2. Record or Tuple

Each row of a table is known as record. It is also known as tuple.

For example, the following row is a record that we have taken from the above table.

1	A	17	abc
---	---	----	-----

3. Field or Column name or Attribute

The above table “STUDENT” has four fields (or attributes): Id, Name, age & Address.

4. Domain

A domain is a set of permitted values for an attribute in table. For example, a domain of month-of-year can accept January, February,...December as values, a domain of dates can accept all possible valid dates etc. We specify domain of attribute while creating a table.

An attribute cannot accept values that are outside of their domains. For example, In the above table “STUDENT”, the Id field has integer domain so that field cannot accept values that are not integers for example, Id cannot has values like, “First”, 10.11 etc.

5. Keys

A relation key is an attribute which can uniquely identify a particular tuple(row) in a relation(table).

6. Schema

A relation schema describes the structure of the relation, with the name of the relation(name of table), its attributes and their names and type.

7. Instance:

The data stored in database at a particular moment of time is called instance of database. Database schema defines the variable declarations in tables that belong to a particular database; the value of these variables at a moment of time is called the instance of that database.

b. Constraints in DBMS

Constraints enforce limits to the data or type of data that can be inserted/updated/deleted from a table. The whole purpose of constraints is to maintain the **data integrity** during an update/delete/insert into a table.

Types of constraints

- NOT NULL
- UNIQUE
- DEFAULT
- CHECK

- Domain constraints
- Entity integrity constraint
- Key Constraints
- Referential integrity constraints

1. NOT NULL:

NOT NULL constraint makes sure that a column does not hold NULL value. When we don't provide value for a particular column while inserting a record into a table, it takes NULL value by default. By specifying NULL constraint, we can be sure that a particular column(s) cannot have NULL values.

2. UNIQUE:

UNIQUE Constraint enforces a column or set of columns to have unique values. If a column has a unique constraint, it means that particular column cannot have duplicate values in a table.

3. DEFAULT:

The DEFAULT constraint provides a default value to a column when there is no value provided while inserting a record into a table.

4. CHECK:

This constraint is used for specifying range of values for a particular column of a table.

When this constraint is being set on a column, it ensures that the specified column must have the value falling in the specified range.

5. Domain Constraint

- Domain constraint defines the domain or set of values for an attribute.
- It specifies that the value taken by the attribute must be the atomic value from its domain.

Example-

Consider the following Student table-

STU_ID	Name	Age
1	John	20
2	Abby	21
3	Ruby	A

Here, value 'A' is not allowed since only integer values can be taken by the age attribute.

6. Key constraints:

Key constraint specifies that in any relation-

- All the values of primary key must be unique.
- The value of primary key must not be null.

Example-

Consider the following Student table-

<u>STU_ID</u>	Name	Age
1	John	20
1	Abby	21
3	Ruby	20

This relation does not satisfy the key constraint as here all the values of primary key i.e. STU_ID are not unique.

7. Entity Integrity Constraint:

- Entity integrity constraint specifies that no attribute of primary key must contain a null value in any relation.
- This is because the presence of null value in the primary key violates the uniqueness property.

Example-

Consider the following Student table-

<u>STU_ID</u>	Name	Age
1	John	20
	Abby	21
3	Ruby	20

This relation does not satisfy the entity integrity constraint as here the primary key STU_ID contains a NULL value for row 2.

5. Referential Integrity Constraint-

- This constraint is enforced when a foreign key references the primary key of a relation.
- It specifies that all the values taken by the foreign key must either be available in the relation of the primary key or be null.

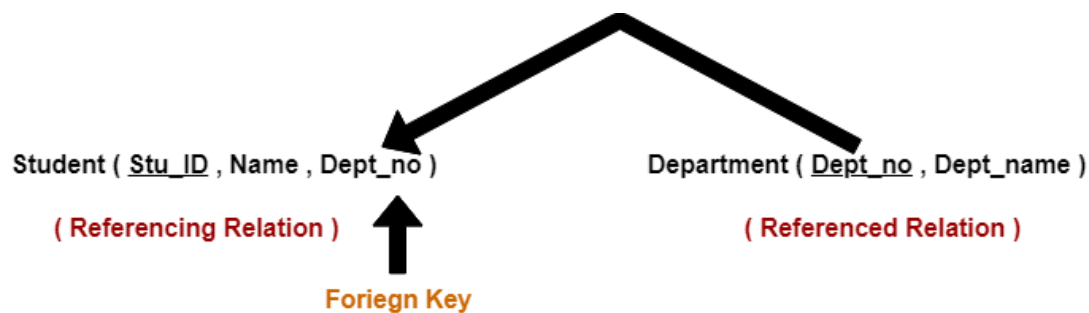
The following two important results emerges out due to referential integrity constraint-

- We can not insert a record into a referencing relation if the corresponding record does not exist in the referenced relation.
- We can not delete or update a record of the referenced relation if the corresponding record exists in the referencing relation.

Example-

Consider the following two relations- 'Student' and 'Department'.

Here, relation 'Student' references the relation 'Department'.



Student Table		
<u>STU_ID</u>	Name	Dept_no
1	John	D10
2	Abbey	D10
3	Ruby	D11
4	Alisha	D14

Department table	
<u>Dept_no</u>	Dept_name
D10	ASET
D11	ALS
D12	ASFL
D13	ASHS

Here,

- The table 'Student' does not satisfy the referential integrity constraint.
- This is because in table 'Department', no value of primary key specifies department no. 14.
- Thus, referential integrity constraint is violated.

c. Different Types Of Keys in DBMS-

- Super key
- Candidate key
- Primary key
- Foreign key

1. Super Key-

- A super key is a set of attributes that can identify each tuple uniquely in the given relation.
- A super key is not restricted to have any specific number of attributes.
- Thus, a super key may consist of any number of attributes.

Example-

Consider the following Student schema-

Student (roll , name , gender , age , address , class , section)

Given below are the examples of super keys since each set can uniquely identify each student in the Student table-

- (roll , name , gender , age , address , class , section)
- (class , section , roll)
- (class , section , roll , gender)
- (name , address)

NOTE-

All the attributes in a super key are definitely sufficient to identify each tuple uniquely in the given relation but all of them may not be necessary.

2. Candidate Key-

A set of minimal attribute(s) that can identify each tuple uniquely in the given relation is called as a candidate key.

Example-

Consider the following Student schema-

Student (roll , name , gender , age , address , class , section)

Given below are the examples of candidate keys since each set consists of minimal attributes required to identify each student uniquely in the Student table-

- (class , section , roll)
- (name , address)

NOTES-

- All the attributes in a candidate key are sufficient as well as necessary to identify each tuple uniquely.
- Removing any attribute from the candidate key fails in identifying each tuple uniquely.
- The value of candidate key must always be unique.
- The value of candidate key can never be NULL.
- It is possible to have multiple candidate keys in a relation.
- Those attributes which appears in some candidate key are called as **prime attributes**.

3. Primary Key-

A primary key is a candidate key that the database designer selects while designing the database.

NOTES-

- The value of primary key can never be NULL.
- The value of primary key must always be unique.
- The values of primary key can never be changed i.e. no updation is possible.
- The value of primary key must be assigned when inserting a record.
- A relation is allowed to have only one primary key.

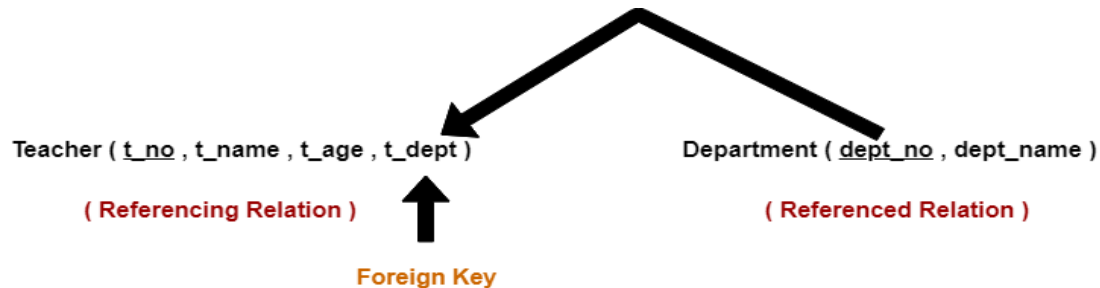
4. Foreign Key-

- An attribute 'X' is called as a foreign key to some other attribute 'Y' when its values are dependent on the values of attribute 'Y'.
- The attribute 'X' can assume only those values which are assumed by the attribute 'Y'.

- Here, the relation in which attribute 'Y' is present is called as the **referenced relation**.
- The relation in which attribute 'X' is present is called as the **referencing relation**.
- The attribute 'Y' might be present in the same table or in some other table.

Example-

Consider the following two schemas-



Here, `t_dept` can take only those values which are present in `dept_no` in Department table since only those departments actually exist.

NOTES-

- Foreign key references the primary key of the table.
- Foreign key can take only those values which are present in the primary key of the referenced relation.
- Foreign key may have a name other than that of a primary key.
- Foreign key can take the NULL value.
- There is no restriction on a foreign key to be unique.
- In fact, foreign key is not unique most of the time.
- Referenced relation may also be called as the master table or primary table.
- Referencing relation may also be called as the foreign table.