|  |
| --- |
| Constraints: Rules defined on the table which cannot be violated by the users. Types: Domain Integerity Constraints: Check: Limits the range of values or domain, checks the values which a col can have. Referntial Integrity Constraints: Foreign Key: Referntial integrity b/w 2 tables, one table col values based on other table col values. Entity Integrity Constraints: Default Constraint: defines the default value for a column. Null: allows null value. NOT null:Donot allow null value. Primary Key: Ensure no duplicate data and not allow nulls. Unique Key: donot allow duplicate values, but allows one null per col.Can have multiple unique constraints on a table |
| There are a number of different ways to implement constraints, but each of them falls into one of these three categories: entity, domain, and referential integrity constraints.  Domain Constraints: A Domain constraint deals with one or more columns. It is important to ensure that a particular column or a set of columns meets particular criteria. When you insert or update a row, the constraint is applied without respect to any other row in the table. The focus is on the data that is in the column. These kinds of constraints will resurface when we deal with Check constraints, Default constraints and rules and defaults.  Entity Constraints: Entity constraints are all about individual rows. This constraint ignores the column as a whole and focuses on a particular row. This can be best exemplified by a constraint that requires every row to have a unique value for a column or a combination of columns. This is to ensure that for a particular row, the same value does not already exist in some other row. We’ll see this kind of constraint in dealing with Primary key and Unique constraints.  Referential Integrity Constraints: Referential integrity constraints are created when a value in one column must match the value in another column. It can either be in the same table or more typically, a different table. For example, we are taking orders for a product, and we accept credit payment. But we will accept only a few standard credit card companies like Visa, MasterCard, Discover, and American Express. Referential integrity constraints allow us to build what we would call a domain table. A domain table is table whose sole purpose is to provide a limited list of acceptable values. In our case we have a CreditCard table with CreditCardID, and CreditCard as fields. We can then build one or more tables that reference the CreditCardID column of our domain table. With referential integrity, any table that is defined as referencing our CreditCard table will have to have a column that matches up to the CreditCardID column of our CreditCard table. For each row we insert into the referencing table, it will have a value that is in our domain list. We will see more of this when we learn about Foreign key constraints.  With Regards, |