DATE / /

C4 *Q. Write any short notes on Constraints in SQL

- Domain Constraints allow us to test whether the values inserted into the database are correct o not.
- CREATE TABLE Command may also include domain constraints which can check integrity of database.
- domain constraints are the most basic form of integrity constraints.

Types of Domain Constraints.

1. Required data constraints / Nullness constraint
2. Check constraint
3. Default keyword.

1. Required data constraint / Nullness Constraint.
- The database may have some attributes mandatory like user registration must have user email address
- These attributes in a database are not allowed to contain NULL values or blanks.
- Ex:
In the student table, student must have an associated student name.
Student_Name varchar(100) NOT NULL
Therefore now, it is a required data column. It is not possible to insert NULL value in that column.
- The DBMS can prevent user from inserting null values with the help of such constraints.

2. check constraints.

- It is used to ensure that attribute value satisfies specific condition as specified by data requirements or user.
- Suppose in Student Table, gender of student can be male or female only.
- The DBMS can prevent user from entering incorrect or other database in db.
- Example:

Table with student entity having gender which can b M or F.

Hence, attribute gender can only take two values either F or M.

Student_Gender varchar(1) (check (gender IN('M','F')).

3. Default keyword.

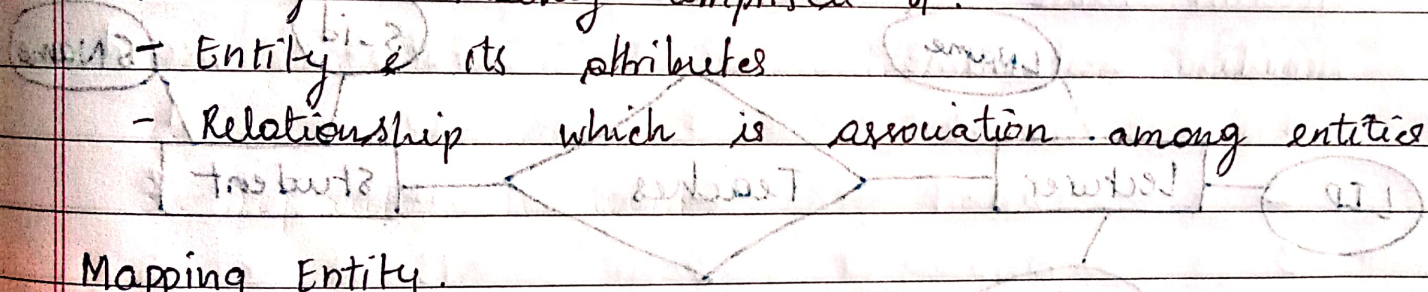
- It is used to add a default specified value, if attribute value is not provided by user.
- It avoids the addition of NULL value to the db by inserting default value as specified by the developer while creating a table.
- Example:

Table with customer entity having name & gender
- if name not added for customer that will be taken as "Unknown" if we specify DEFAULT value of NAME column to "Unknown".

STUDENT_NAME VARCHAR(50) DEFAULT "UNKNOWN".

* Q. Explain the steps of an algorithm for ER to relational mapping.

- ER Model when conceptualized into diagrams gives a good overview of entity relationship, which is easier to understand.
- ER diagrams can be mapped to Relational schema using step by step procedure.
- Though all the ER constraints cannot be imported into Relational model, but an imported approximate schema can be generated.
- ER diagrams mainly comprised of:
  - Entity & its attributes
  - Relationship which is association among entities.



Mapping Entity.

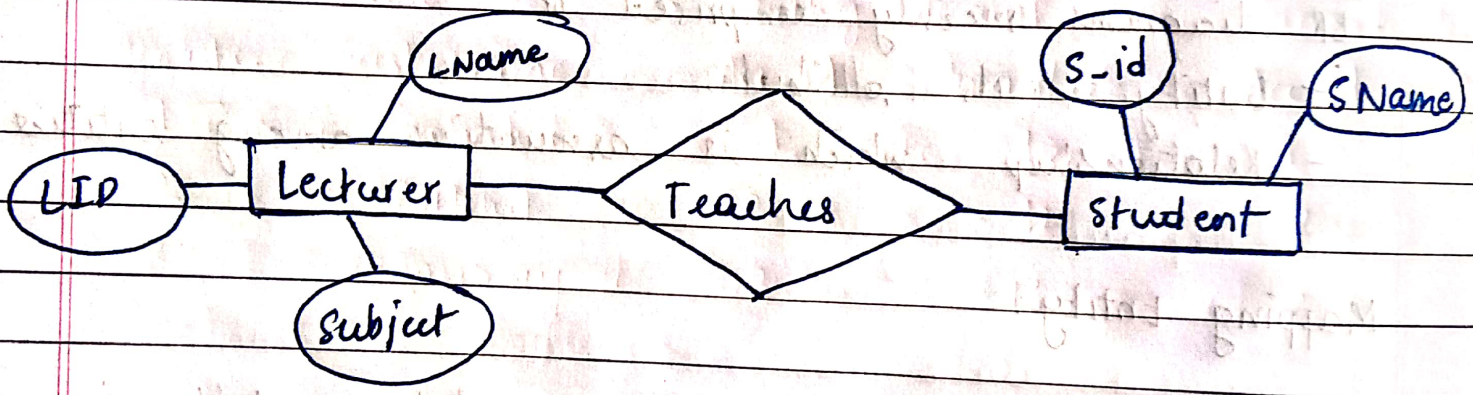
- Regular entity sets can be represented as table in relational model.
- Attributes of entity set can be converted to the columns (attributes) of the tables.

- Create table for each entity
- Entity attributes should become fields of tables with their respective data types.
- Declare primary key.

DATE / /

Mapping relationship

- Relationship is among entities
- Create table for relationship
- Add the primary key of all participating Entities as field of table with their respective datatypes
- If relationship has any attributes, add each attribute as field of table
- Delare a primary key composing all the primary key of participating entities
- Delare all foreign key constraink



Lecturer table

Lid	LName	Subject
1	Amit	IP
2	Ayaan	INS
3	Jay	PM

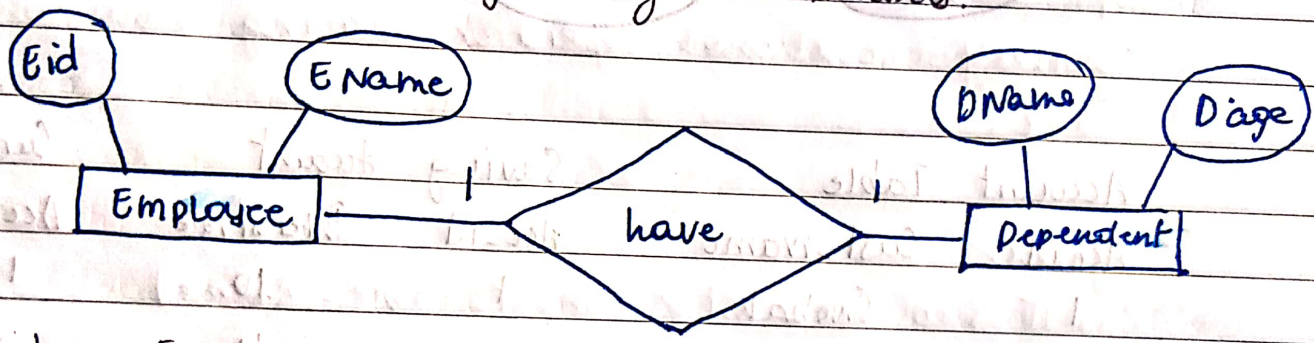
Student Table

Sid	SName	Lid
1	Aarav	1
2	Sneha	1
3	Maya	2
4	Kabir	3
5	Siya	3
6	Rahul	2

Mapping Weak Entity sets.

A weak entity is one which does not have any primary key associated with it.

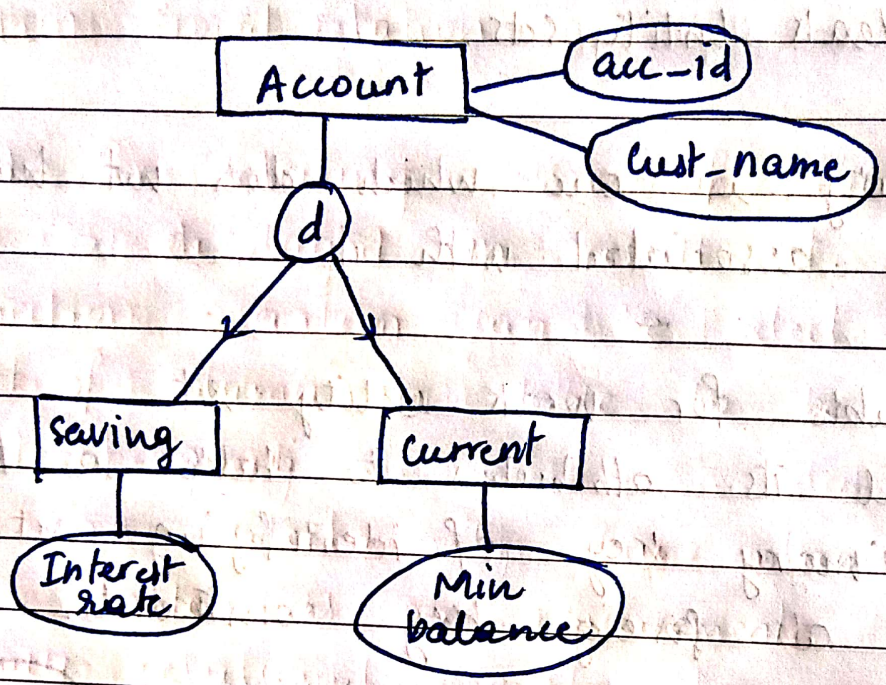
- Create table for weak entity set
- Add all its attributes to table as field
- Add primary key of identifying set
- Declare all foreign key constraints.



Eid	Ename	Dname	DAge
1	Sachin	Jyoti	23
2	Suhas	Manju	24
3	Shekar	Tej	27.

Mapping hierarchical entities

- ER specialization or generalization comes in the form of hierarchical entity sets.
- Create tables for all higher level entities
- Create tables for lower level entities
- Add primary keys of higher level entities
- In lower level table, add all attribute of lower level entities
- Declare primary key of higher level table to primary key for lower level tables.



Account Table

Acc id	Cust_name
1	Snehal
2	Nikhil
3	Janmay

Saving Account

Acc_id	Interest rate
1	10
2	12
3	13

Current Account

Acc id	Min
1	1000
2	2550
3	1150

MIII C5 * Q. Explain select operation (σ)

- operator is used to select some rows from table which satisfies particular selection condition given in selection operation
- operator selects a set of tuples that satisfy a selection condition
- output of query is exactly same as input of table
- this is unary relational operator having only one input table.

Syntax:
- σ <attribute_name> < comparison_operator> < constant_value> (Input_table_name).

Where,
Attribute_name : Name of column in table
Comparison_operator : =, <, >, <=, >=, <>
- ex:
  Query : Select all employees having age below 30 years.
  Solution : σ age < 30 (Employee)

Combining multiple conditions
- We can have more than one condition by using logical connectives like AND ($\wedge$), OR ($\vee$).
- Query: Select all employees having salary above 5000 and age above 65.
- Solution : σ salary > 5000 $\wedge$ age > 65 (Employee).

* Q. Explain Project relation

- used for selecting some of many column in table to displayed in result set.
- projection operation can be select a column or a set of column in table to be displayed in output of query.
- we can select only few columns or all colum of a table as per requirements.
- unary relational operator having only one input table.

Syntax:

Π < column_list > (Input_Table_Name)

Ex: Query: Find salary and age of all employees.

Solution: Π age, salary (Employee).

*Q. Rename operation

- We can give alternative name to any column or any table of query expressions using operator called as RENAME operator.
- This operator is specially introduced to select specific column from joined table containing multiple columns of same name column
- rename operator, denoted by the lowercase Greek letter (ρ)

Syntax:

ρ (New_name) (Input_Table_name)

Example:

(ρe (Employee)).

* Q Set Union

- operation. Find out all combined rows in table1 and table 2
- union effectively appends the result of first query to the result of second query.
- It does not eliminate all duplication rows and they are printed in result expression.

Syntax

(Query Expression 1) U (Query Expression 2)

IT Table

Eid	Ename	Age
11	Jay	24
12	Suhas	25
13	Sachin	23
14	Harsha	25

Computer table

Eid	Ename	Age
21	Varsha	24
22	Bhavna	24
23	Ekta	25
24	Amrita	23

Query: Find all employees in computer and IT departmen
Solution: (IT_Employee) ∪ (Comp_Employee)

2. Explain Cartesian product.

- A cross join performs relational product or cartesian product of two tables specified in query.
- in this case every row in first table will be joined with every row in second. So finally number of rows in result table will be equals to product of number of rows in table 1 and number of rows in table 2.
- that means all rows in the first table are joined to all rows in the second table.

Syntax:

(Query Expression1) X (Query Expression 2)

Employee

Eid	EmployeeN	Did
1	Jay	100
2	Mahesh	200
3	Suhas	100

Department

Did	Dname
100	HR
200	TIS

Eid	Ename	Did	Did	Dname
1	Jay	100	100	HR
1	Jay	100	200	TIS
2	Mahesh	200	100	HR
2	Mahesh	200	200	TIS
3	Suhas	100	100	HR
3	Suhas	100	200	TIS

Query: Find all combination Employees & Departments.

Cartesian product

- DBMS, cross join occurs due to where condition is missing in a query or some invalid operations in where clause leads to undesired results or CROSS Join.
- A cartesian product is formed when:
  - A join condition is omitted
  - A join condition is invalid

To avoid a Cartesian product, always include a valid join condition in a where clause.

* Q. Join operation ($\bowtie_\theta$)

- help us retrieve data from multiple tables & relations.
- most common type is Natural join.
- Syntax:

  (<table_name1>) $\bowtie$ <join condition> (table_name2).

- 3 types of joins:
  1. Natural join
  2. Inner join / Theta join
  3. Outer Join

1. Natural join

- can join tables based on the common columns in the tables being joined.
- returns all rows by matching values in common column having same name & datatype of column and that column should be present in both tables.

Pre requisite for natural join
Both tables must have at least one common column with same column name and same data.

Steps of working:
- Two tables are joined using Cross join
- DBMS will look for a common column with same name & data.

Example:

Employee

Eid	Ename	Did
1	Amit	10
2	Yogiesh	30
3	Nitin	50

Department

Did	Dname
10	HR
30	TIS
40	IT

Query : Find all employees and their respective departments.

Solution : (Employee) ⋈ (Department)

Eid	Ename	Did	Did	Dname
1	Amit	10	10	HR
2	Yogiesh	30	30	TIS.

- All non matching tuples of cross join are ignored

2. Inner joins / Theta Join ($\bowtie_\theta$)

- will combine tuples from multiple relations if they satisfy the specified join condition
- also called as theta and denoted by θ
- tables are joined according to join conditions.
- Example :

  Query: Find all Employees and their respective departments

  Solution: Employee ⋈ employee.did = Department.did Dep

3) Outer joins:

- In an inner join or in case of simple join, the resultant table contains only the combinations of rows that satisfy the join conditions.
- Rows that do not satisfy the join conditions are discarded.
- Outer join, joins two table although there is no match between the joining table.
- Outer joins are usefull when you are trying to determine which values in related tables in related table cause referential integrity problem.
- Such problems are created when foreign keys values do not match the primary key values in related tables.

i) Left outer join

- Table on left side of operator may contain null values
- Left outer join takes all tuples in the left relation that did not match with any tuples in right relation.

Query: Find all employees and their respective data

Solution: Employee ⟕ employee.id = department.did.
Department

refer. Techmax p. 5-15

ii) Right outer join.

- Table on right side of operator may contain null values
- Right outer join takes all tuples in the right ~~direction~~ relation that did not matche with any tuple in the left relation.

Query: Find all departments with employee data.
Solution: Employee $=\bowtie$ employee.did = Department.did Department.

iii) Full outer join

- any table on both sides of operator may contain null values.
- Query: Find all Employees and departments
Solution: Employees $=\bowtie=$ employee.did = Department.did Department.