Relational DB Model

This model consist of no. of tables called as relation where each relation will have a unique name. Each relation consist of no. of rows called tuples and no. of columns called attributes.

Domain – Set of permitted value for an attribute
ex - datatype, size.

Degree of relation – no. of attribute present in any relation.

Cardinality of relation – no. of tuples present in any relation.

Rule1 – The attribute values of a relation in Relational DB model is required to be atomic i.e. indivisible.
i.e. No composite attribute is allowed.

Database Scema – Logical design of a database without having any data. ex - definition of table.

State of relation / Database instance – It is the snapshot of data in database at any particular time. Represented by r(R) or s(R).

Defining a relation –

(i) R(a1 datatype1 (constraint if any), a2 datatype2 (constraint if any), ...)

(ii) $R \subseteq dom(A1) \times dom(a2) \times \dots \times dom(An)$

↳ max^m no. of tuples in a relation $|dom(a_1)| \times |dom(a_2)| \times \dots \times |dom(a_n)|$

Constraints Types

(I) Model based or Implicit Constraint

ER - you can't represent cardinality in recursive relationship.

Relational model - It can't have duplicate rows i.e. two tuples having exact value for each attribute.

- you can have value only atomic.

(II) Schema based or explicit Constraint

Types of Schema based / explicit

i) Domain Constraints - This constraints specify within each tuple the value of each & every attribute 'A' must be an atomic value from the DOM (A).

ex - Number (2) : 'F' X

'3' ✓

133 X

ii) Integrity Constant - No two tuples in a relation can have same set of values for each attribute.

$t_1[A_1, A_2 \ldots A_n] \neq t_2[A_1, A_2, \ldots A_n]$

Key Constraints – Every relation in a DB must have a key (set of one of more attribute) that can identify a type uniquely in the relation and key can't have null value.

Candidate key – A set of attribute is called as candidate key is they satisfy the following properties.

(a) Uniqueness – Two tuples in any relation can not have identical values for all the attributes in the key.

ex choose A or B

(b) Minimality – No proper subset of the key should satisfy uniquess property.

ex If AB is sufficient to uniquely identify all tuples then ABC or ABD won't be considered as candidate key.

iii) **Referencial Integrity** – This constraints state that if there is a foreign key in one relation then each foreign key value must match the primary key in other table (or the foreign key value can be null).

→ Maintain consistency among the rows of the relation.

ex

Student (F.K)

Student_name	Sname	Fid
101	a	F1
102	b	F2
103	c	F6 (X)
104	d	-

Faculty (P.K)

Fid	F_name
F1	A
F2	B
F3	C
F4	D

as not in

★ If F1 | A deleted then we also need to delete 101 | a | F1 first

A foreign key that references it's own relation called recursive foreign key.

ex

Eid	Manager-id
E1	E3
E2	E3
E3	–
E4	E2

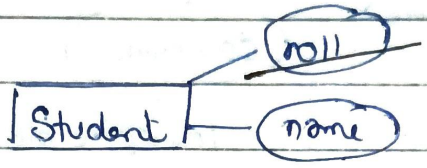
(iii) Application based or semantic

ex

For salary we need to execute negative value.

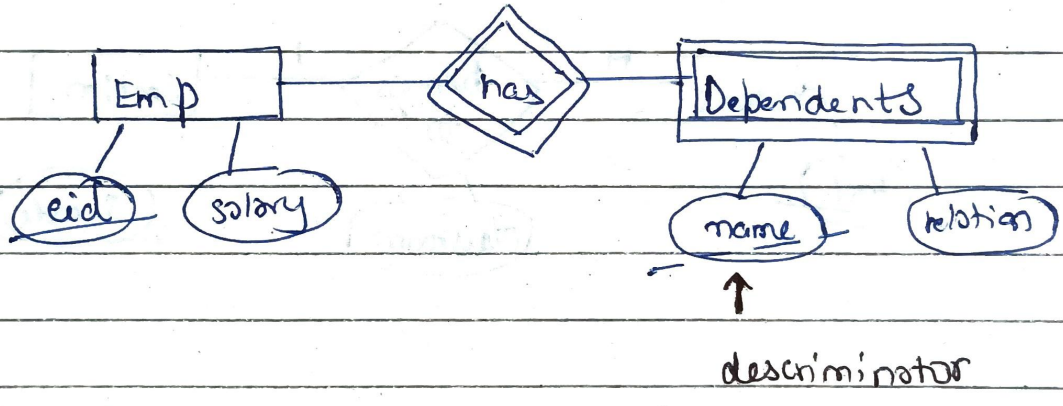
Conversion of ER model to relational model

(i) Strong entity



Student = { roll , name }
PK

(ii) Weak entity

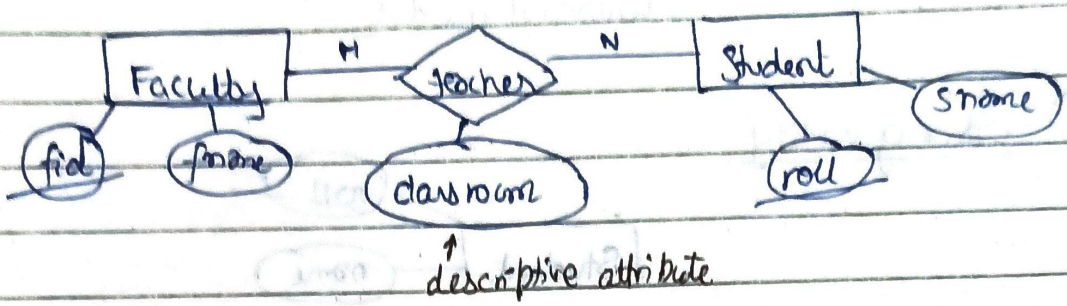


Emp = { eid , salary }
PK

Dependents = { eid , name , relation }
P.K

- need to refer the P.K of strong entity for describing weak entity.

(III) M:N relationship

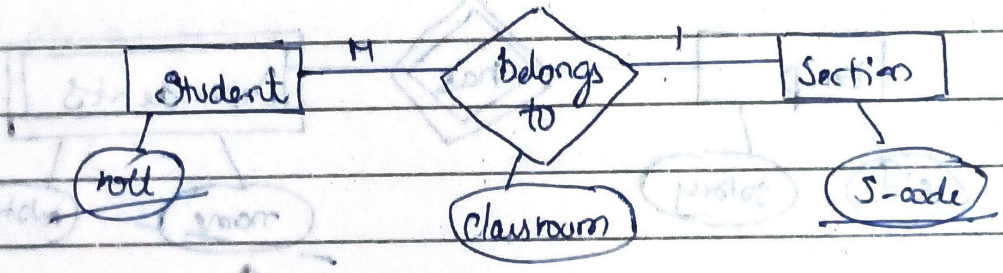


Faculty = { ---- }

Student = { ---- }

teaches = { $\underset{FK}{\underline{fid}}$, $\underset{FK}{\underline{roll}}$, $\underset{\text{descriptive attribute}}{\underline{Classroom}}$ }

(IV) 1:M relationship

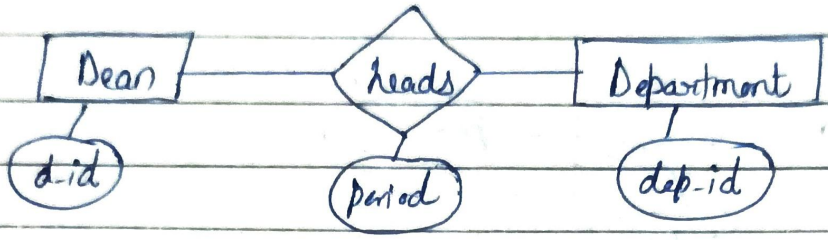


Student = { $\underset{PK}{\underline{roll}}$, ..., $\underset{FK}{\underline{S\text{-}code}}$, $\underset{\text{descriptive attribute}}{\underline{classroom}}$ }

Section = { $\underset{PK}{\underline{S\text{-}code}}$ ---- }

- The PKey of one side entity and descriptive attribute are referred to many side.

v) 1:1 relation



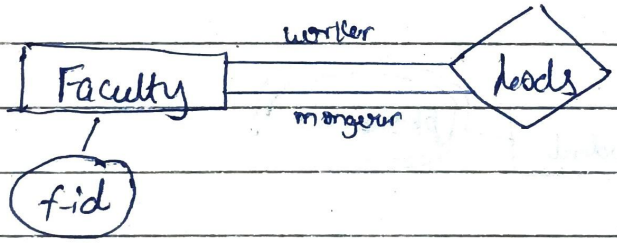
- Assume any one side to be many and then distribute like (iv).

Assume Dean to be many side.

∴ Dean = { did ——, dep-id, period }
PK PK

Department = { dep-id, —— } descriptive attribute
PK

vi) Recursive relation

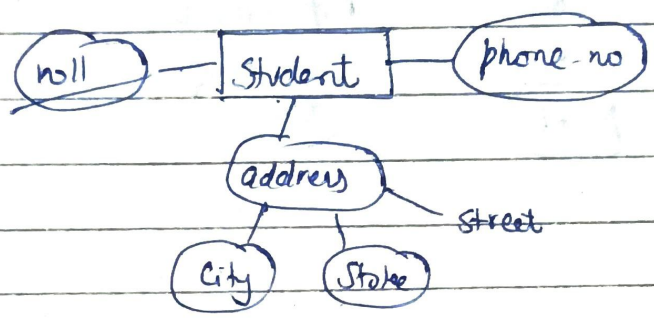


Way 1 → Faculty = { fid, ..., head-id }
PK FK

Way 2 → Faculty = { fid, —— }
PK

Heads = { fid head-id }
PK

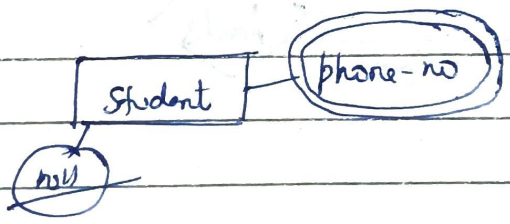
viii) Composite attribute



Student = { roll , ---- add-city , add-state , add-street }
PK

- no need to include 'address' in this.

ix) Multivalued attribute

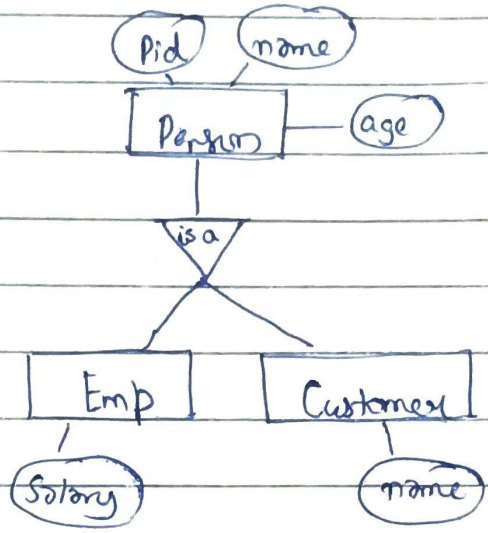


Student = { roll , ---- }
PK

- no need to include phone-no in it.

student-phone-no = { roll , phone }
PK

x) Specialization



person = { $\underline{\text{pid}}_{\text{PK}}$, name, age }

Emp = { $\underline{\text{pid}}_{\text{FK}}$, Salary }

Customer = { $\underline{\text{pid}}_{\text{FK}}$, Name }

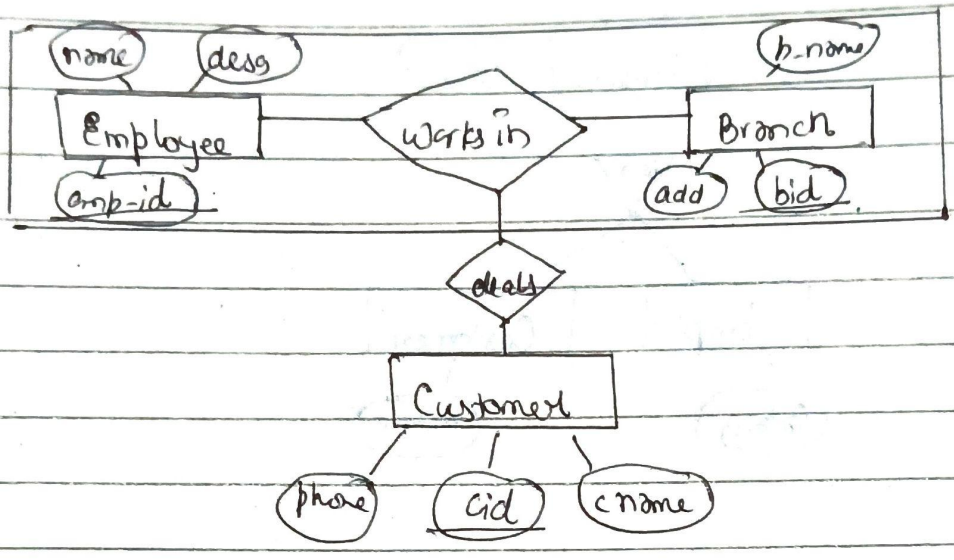
★ If specialization is

(1) disjoint then table for super class is not required.
So it will be like —

Emp = { p_id, name, age, salary }

Customer = { p_id, name, age, name }

xii) Aggregation



Emp = { emp-id, name, desg -- }
PK

Branch = { bid, b_name, add -- }
PK

Customer = { cid, cname, phone }
PK

Works in = { emp-id, bid }
PK

deals = { emp-id, bid, cid }
PK

- **Key** – It is a set of one or more attributes that describes an entity uniquely in the entity set.

ex Roll-no
Transaction ID

Types of Keys –

a) **Super Key** – Set of one or more attributes that describes an entity uniquely called Super Key.

☆ It can be any set of attributes.

ex Roll-no
Phone-no X
Name and DOB
All attributes too.

b) **Candidate Key**

Minimal of Super key.
ex Roll-no in Roll no & Name
Name and address in Name, Add, dob.

c) Primary key

The candidate key which is choosen by DBA.

ex Roll-no or Name, Add

and all other's called Alternate keys.

d) Surrogate key

when none of the attributes can be choosen as a primary key instead of choosing a composite key. DBA choose an extra column as primary key.

ex Instead of taking 4 attributes together as a primary key we use another column.

e) Foreign key

Primary key in one table that is being referred in another table called foreign key. Mainly done to establish relationship bet: them.

ex

To create relationship bet: 2 or more tables.

Table 1

Name	Roll-no	Sub-id
A	1	X
B	2	Y
C	3	Z
D	4	X

Table 2

Sub-name	sub-id
AB	X
CD	Y
EF	Z
GH	W

Foreign key