* Data abstraction
* Multiple View
    - concurrency → transaction correctly
    - OLTP
* When to use DBMS & when not to.?
* Data Model → set of concepts
* Schema → description

(slide 26)

Three Schema Arch → (Description)

i) Internal
ii) Conceptual
iii) External

Data Models → set of concepts

10.02.23

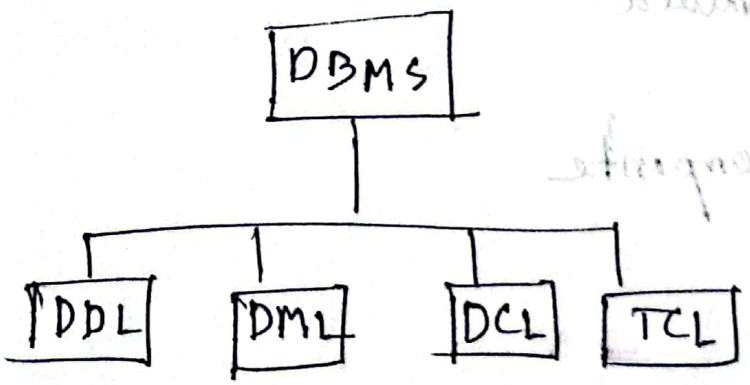
Quiz on Wednesday
ppt of
chap 1, 2

⊛ Data Independence

i) Logical
ii) physical

External
↑
Conceptual
↑
Internal

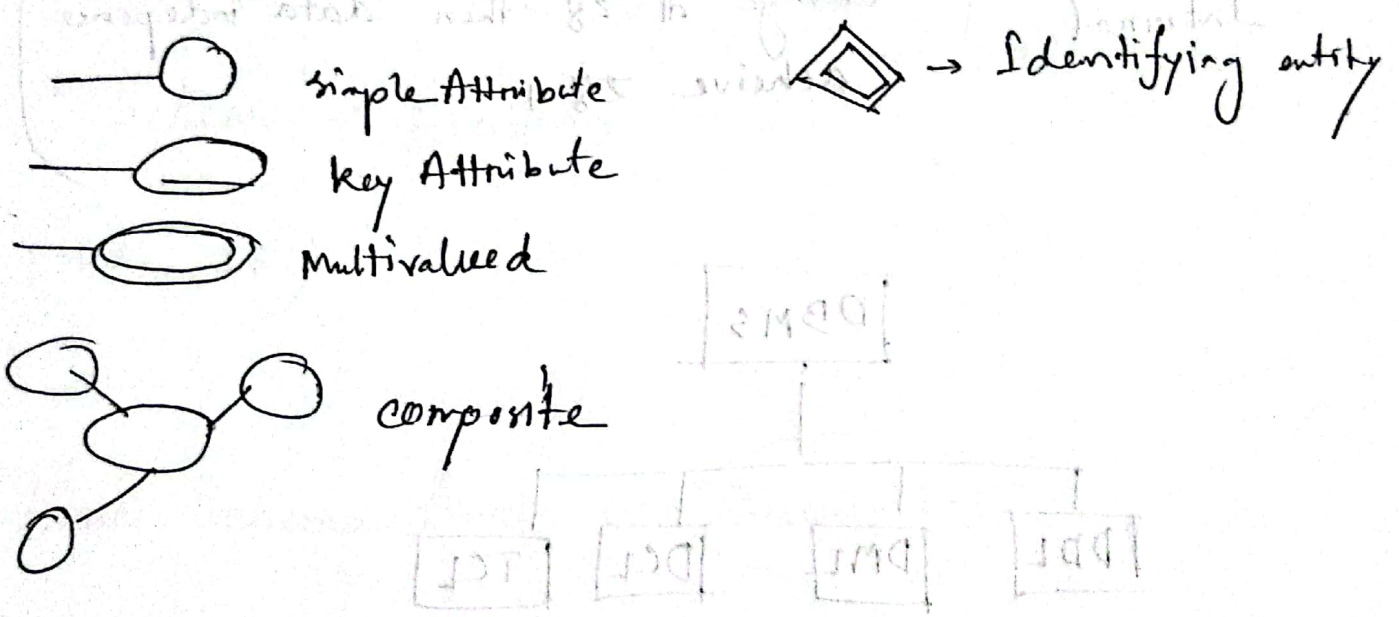
(Change করার জন্য higher level এ change না হয় then data independence acheive হয়।)



Chapter 03

Data Modeling using Entity Relationship (ER)

* Conceptual Design
* entity এর attributes থাকবে



(*) entity এর unique কোন Attribute না থাকলে weak entity

(*) Recursive Relation

(*) Role Assignment

Actors on the Scene

i) DB Administrators

ii) DB Designers

iii) Software Engineers

iv) End Users

1. Casual
2. Naive
3. Sophisticated
4. Stand Alone

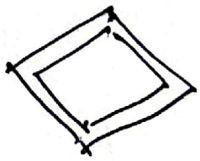
(*) Cardinality → Maximum participation (right-left) (opposite side)

- 1:1
- 1:N
- M:N

(*) Existence Dependency → Minimum participation (যে পাশ্চে line side নেই)

- Single line → zero → optional
- Double line → one → mendatory

(*) Identifying entity



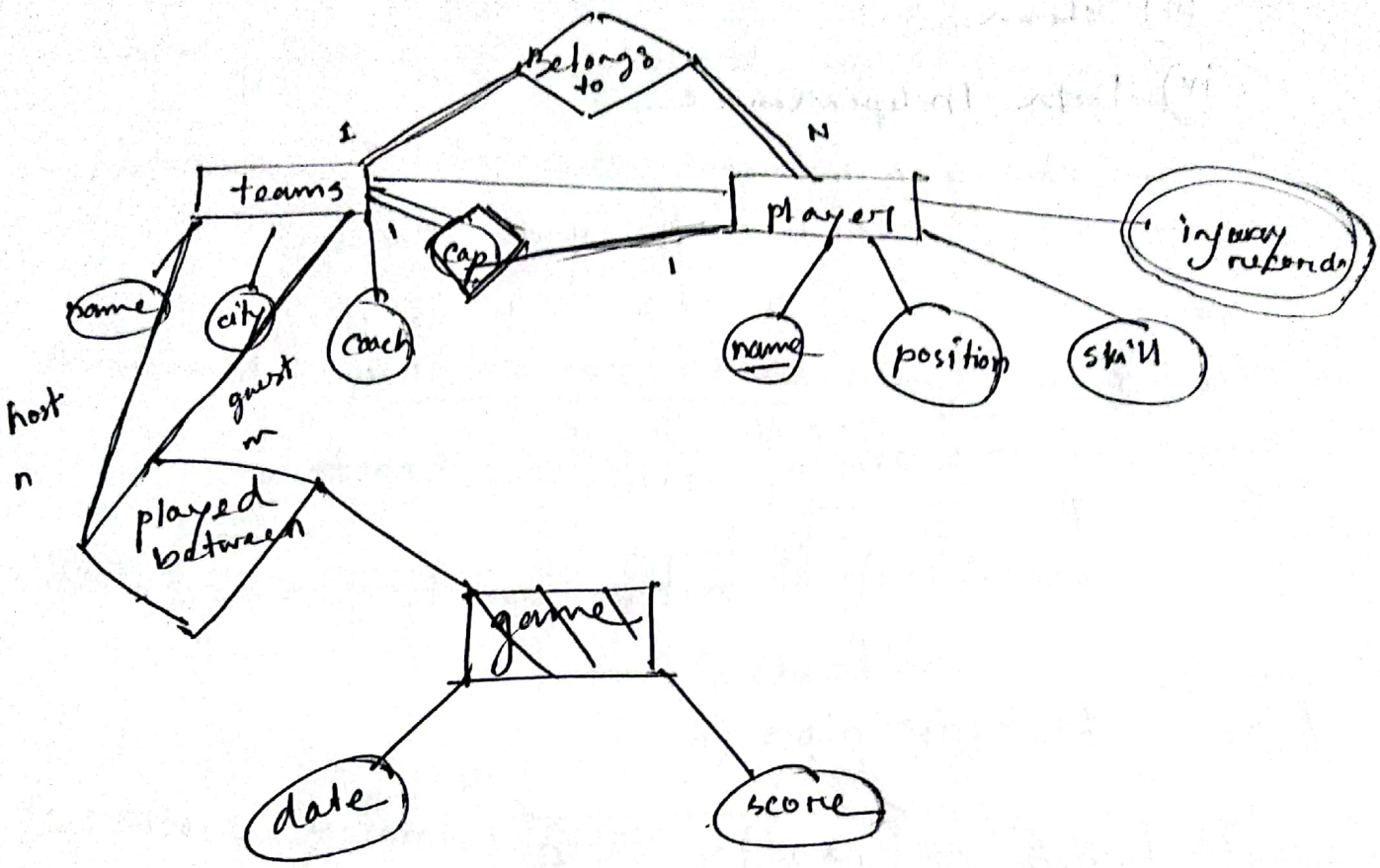
Quiz on short Conceptual Question

- Scenarios
- Suitable উত্তর
- Concept

P01

National Hockey League

ER



Quiz 01
Ultimate

Data : facts to store (that can be recorded), which has implicit meaning

Database: Related data logically coherent built for specific purpose.

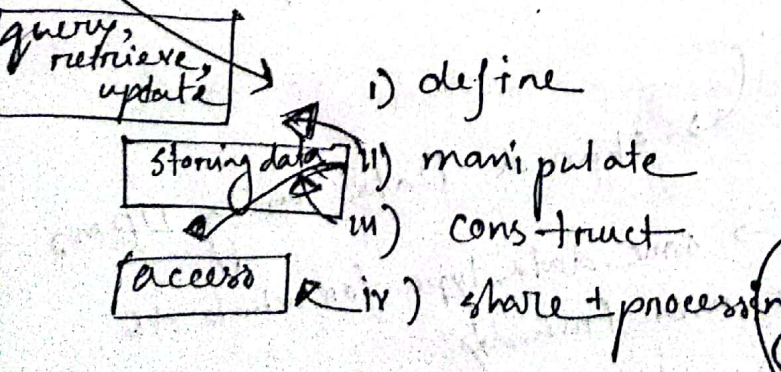
Miniworld: part of real world about which data is stored in DB. Universe of Discourse (UoD)

DBMS: a software system to facilitate
- i) creation
- ii) maintainance

} of computerized DB.

Database System: DBMS software + Data + applications

* an audience is actively interested in its content



Meta Data: The definition / description of DB is stored in form of DB catalog / Dictionary.

A: How other data is described in a DB.

Types of DB

- Traditional
  - Numeric
  - Textual
- Recent
  - Multimedia
  - Real-time & active DB
  - GIS → Geographic Info sys
  - Bio & genome
  - DataWarehouse
  - Mobile Databases

NOSQL → no meta data

* Online Analytic Processing (OLAP) ← Image processing

Data Management:

i File Based Approach
- Manager own data
- → storage waste / common data overlap
- → difficult data maintainance
- → not consistent data
- → data types may be differ

ii DataBase Approach
- → self describing
- → shared Data using DBMS
- → same data type for whole db.
- → user friendly
- → multiple views of data
- → Insulation between programs

Data Abstraction → Program Data Independence

Doesn't require changing structure like fileshare

Application program / user.

↳ query → DB → Manipulation → Read / Write

Retrieve
query
update

Functions of DB:

i) protection
ii) Maintainance

* active processing → triggers, stored procedures
* presentation & visualization

(*) Requirement specifications & Analysis

↳ (I) Conceptual Design → Entity-Relation Ship Model

(II) ↳ Logical Design ⇒ expressed in data model → DBMS implem

(III) ↳ Physical Design - Data input & manipulation

(*) Data Abstraction → provides conceptual view to end users
→ hides storage details

↳ i) program data independence

↳ ii) program operation independence

Characteristics of DB Approach

(1) Self describing nature of DB

→ catalog / Dictionary stores datatypes, structures & constrains of data

this description → meta data

→ allows to work with other DB apps.

(2) Insulation between program & Data (Application program or)

→ the way to access the data remains same even if data format/structure changes

Program Data Independence

(3) Multiple views

(4) Data abstraction

(5) Share: concurrency contro

(6) Recovery

The characteristics that allows

i ↳ program data independence

ii ↳ program operation independence

(*) OLTP → online transactions processing.

(1) isolation

(2) atomicity

Actors on the Scene

* use/control DB content
* design/develop/maintain db applications

Workers Behind the Scene

* computer sys. operators
* design/develop DBMS software/tools
  - tool devs

DBA
- DB Admins → authorizes access to DB/monitor/control
- DB designers → construct → structure/constraints
  communicate w/ end-users → conditions
- Software Engineers
- End users

Constraints

Triggers/Rules → deductive DB Sys

Chap 02

DB System & Concepts Arch

Data Models: A set of concepts to describe the structure of db. (i) operations (ii) constraints

Catagories:

1. Conceptual → high level → semantics → how end users perceive data
2. Physical → low level
3. Implementation → representations

Schemas:

Description of a database → structure

↳ intension → doesn't change frequently

State:

actual data stored in DB at a certain time.

↳ Instance / extension

→ changes very frequently.

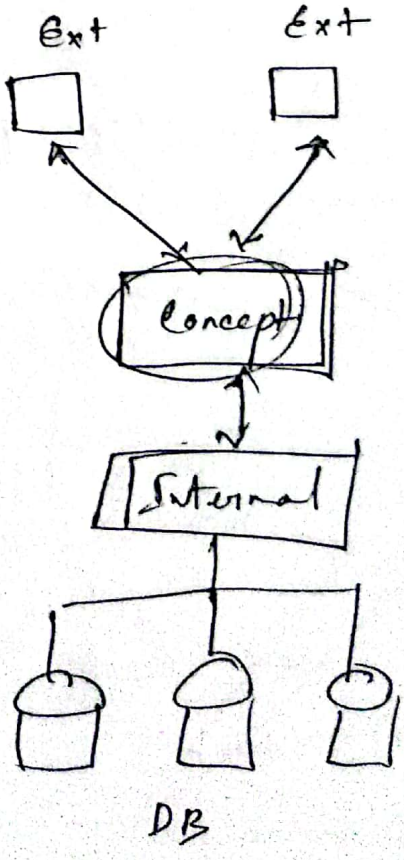
3 Schema Arch :

1) Internal Schema (low level)
   → physical storage, access paths

2) Conceptual Schema (high level, English)
   → describes entity, structure
   → constraints

3) External Schema
   → particular data for end user
   → subset of conceptual

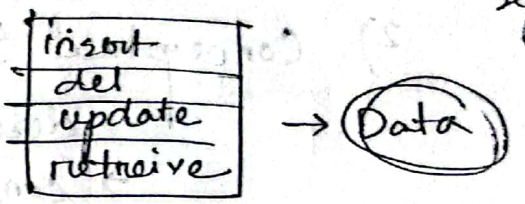
(a) program data independence
(b) multiple views of data



DDL → define DB schemas
↳ construct & store
creation / deletion → table

DML → Manipulate
high lvl : SQL ← complex
low lvl : PL/SQL ← processes seperately

DBMS Interface



DBMS

① DBA staff → Runtime

② Casual → Query compiler → Query Optimizer

③ Application programmers

④ parametic user

Network Model

Hierarchical Data Model (HDM)

(RM) Relational Model

OObject oriented Data Models : OO programming
Object Relational Data Models C++

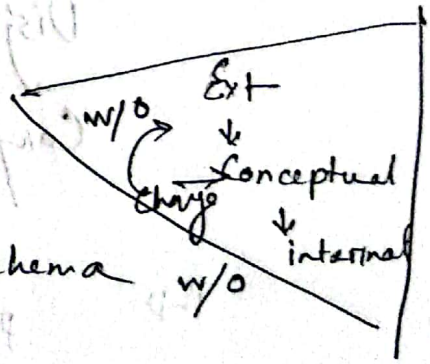
Data Independence:

ability to change the schemat at one level w/o impacting the schema at higher level.

(1) Logical Data Independence :-

adding new col

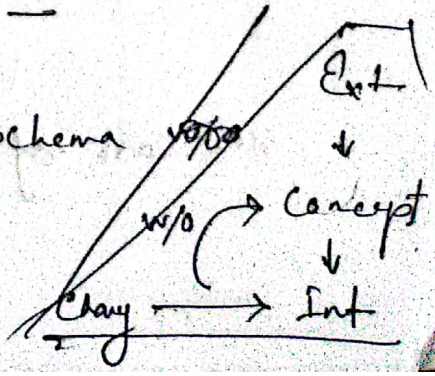
ability to change Conceptual Schema w/o having to chang Ext.



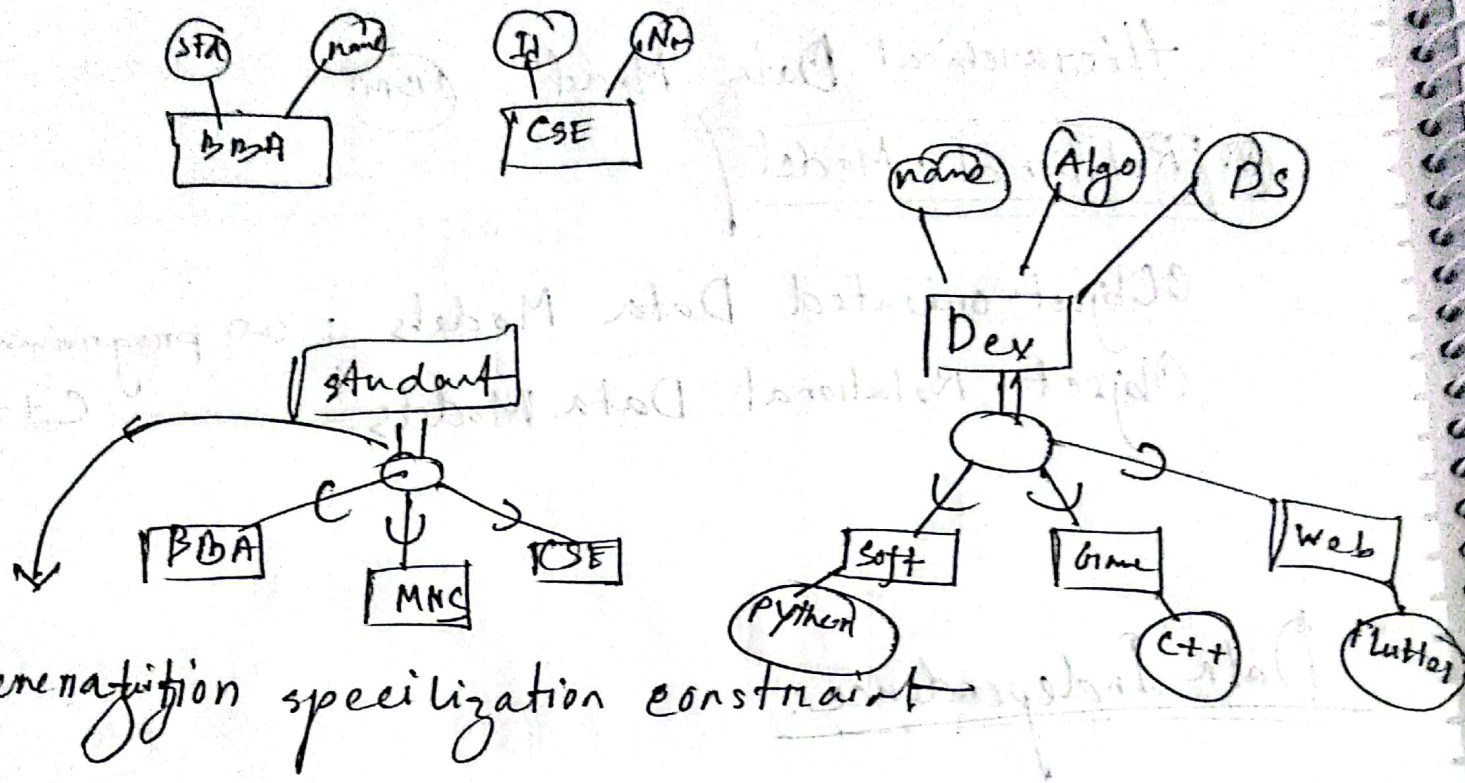
(2) Physical Data Independence :-

eg. structures indexes

Capacity to change internal schema w/o having to change conceptual.



10-11-23

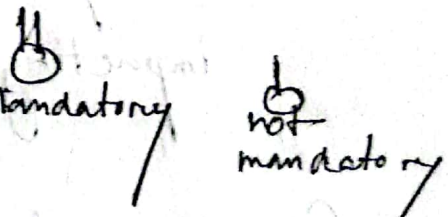


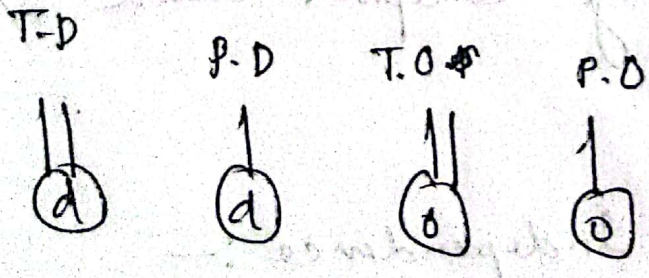
generalization specialization constraint

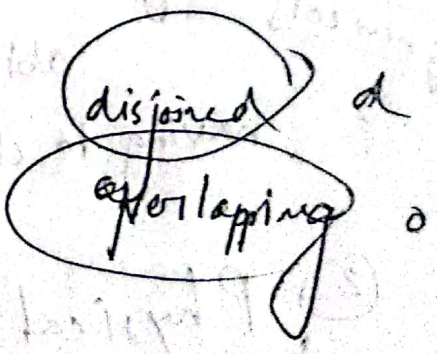
(xo) Specialization:

Disjointness Constraint

Completeness Constrained





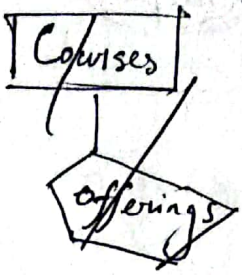


subclass of subclass → hierarchy / Lattice

multiple Super class

10.16.23

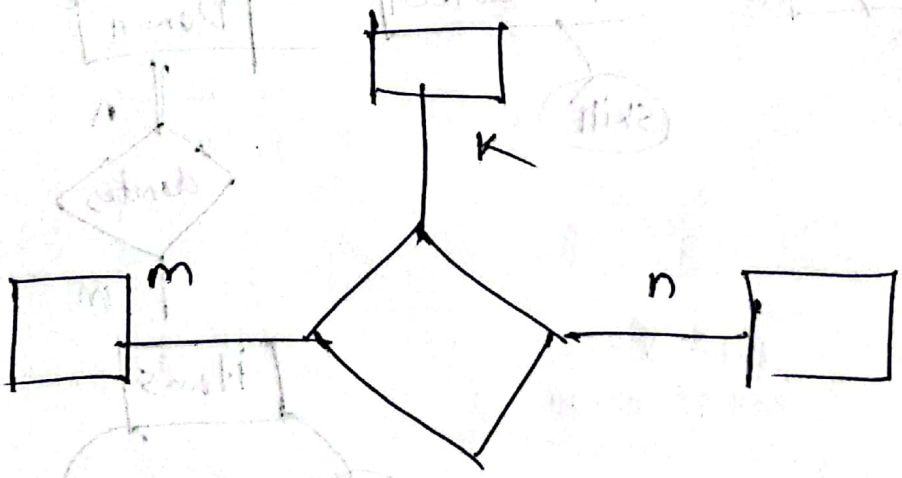
CSE 471



Alex Xu
System Design
2020

System Analysis
Wixom Roth

Ternary Relationship

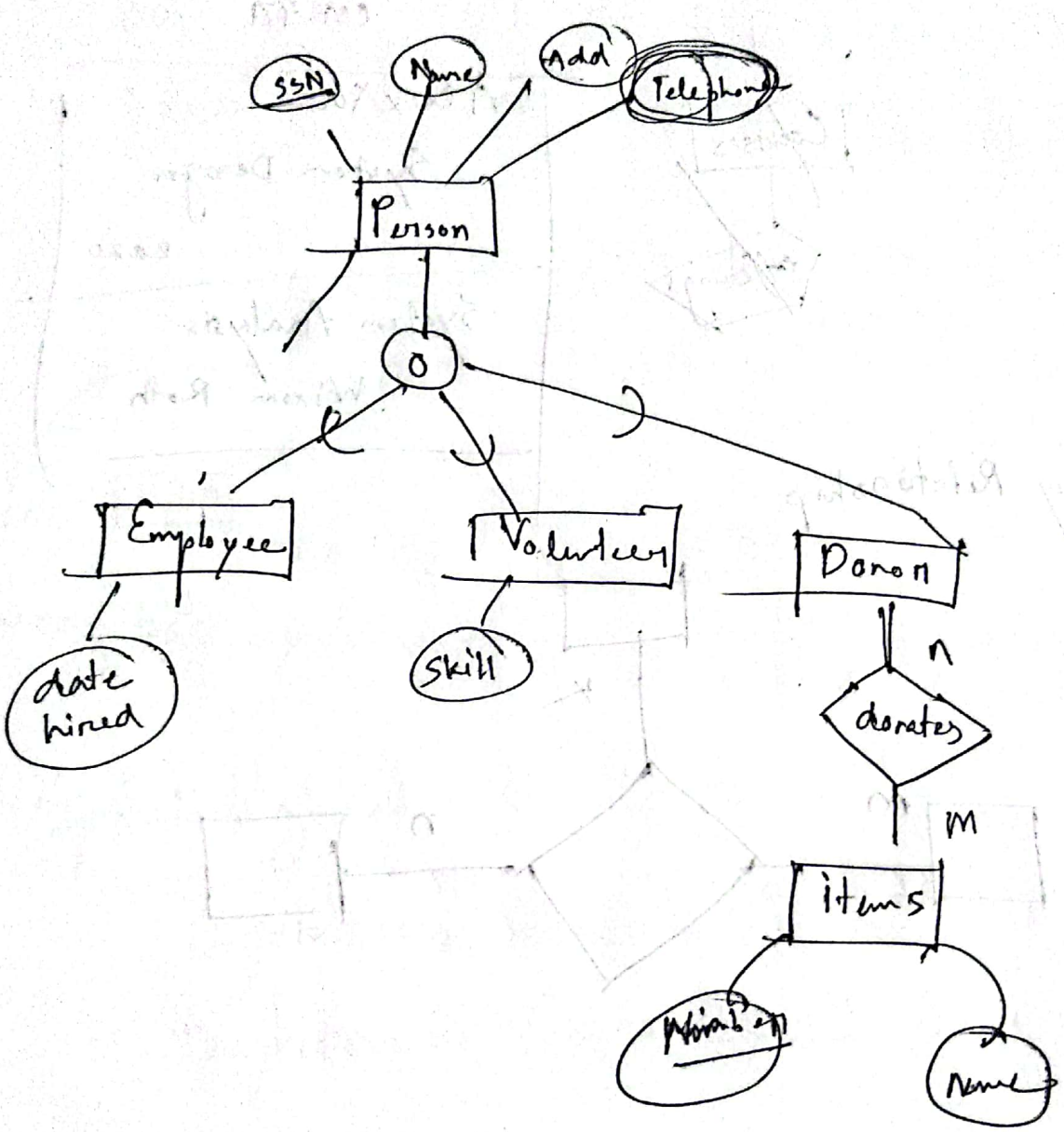


EER

Q1

Ass → 25th

Q03 → 31st

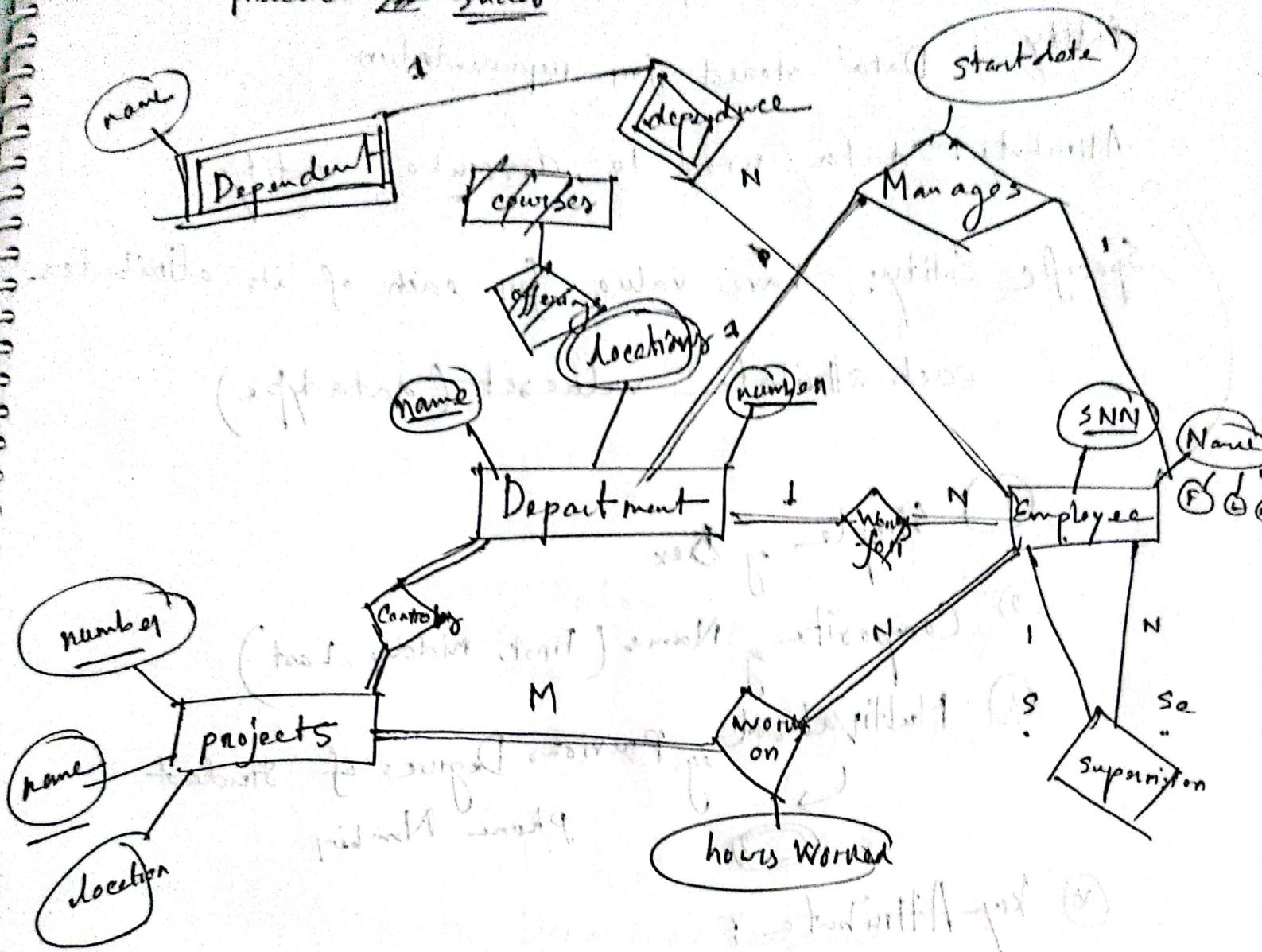


Relationship Attribute key attribute not possible

composite/ multivalued possible

N.b: don't forget Assumptions

E.R. practice ~~#~~ Slides



Chap 03

Entity : Data stored for representation

Attributes: data used to describe entity.

Specific Entity: have value for each of its attributes.

each attribute — valueset (data type)

1. Simple eg Sex
2. Composite eg Name (First, Middle, Last)
3. Multivalued eg. Previous Degrees of Student, Phone Number

4. key Attributes:

each entity must have an unique value

e.g. SNN of Employee

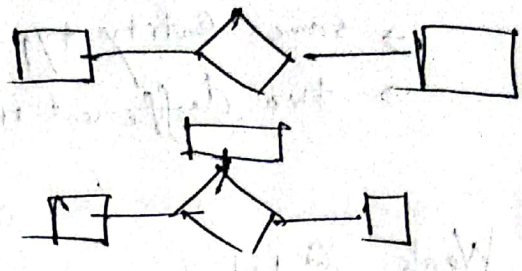
→ (underlined)

(x) Relationships: Degree of Relationship

(i) Binary :

(ii) Ternary :

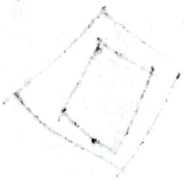
(iii) N-ary :



(x) Constraints

i) Cardinality Ratio :

- 1:1
- 1:N / N:1
- M:N



ii) Existence Dependency Constrain.

- Zero (optional) → single line
- One/more (mandatory participate) → doub

Sup Ssee

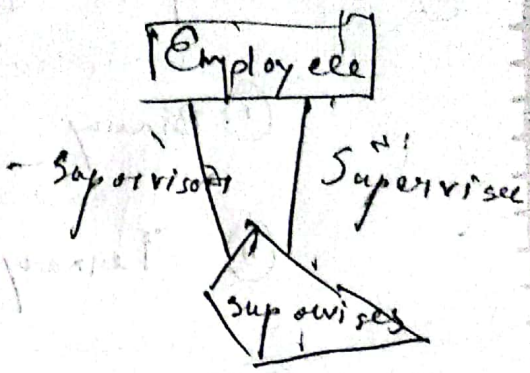
1 → M → M

1 ← 1

(39) Recursive Relation :

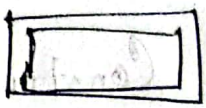
→ same Entity type

→ two different roles

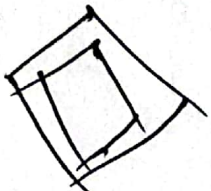


(41) Weak Entity

→ doesn't have key attribute



(42)



1 M

1 1

(43) Identifying Relationship →

connect weak & strong entity

(40) partial key:

combined with the identifying with weak entity can get a unique key.

partial key

* Relationship attribute:

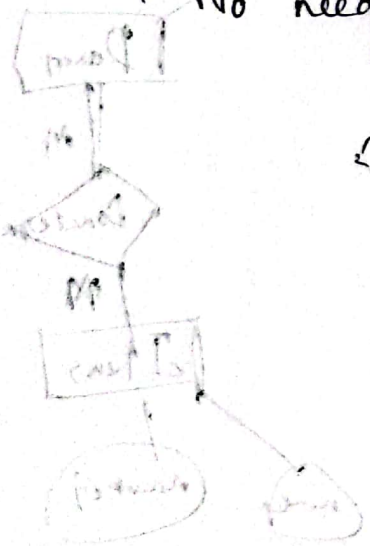
⇒ depends on relationship

e.g. Hours per week of Works_On is a relationship between Employee & Project.

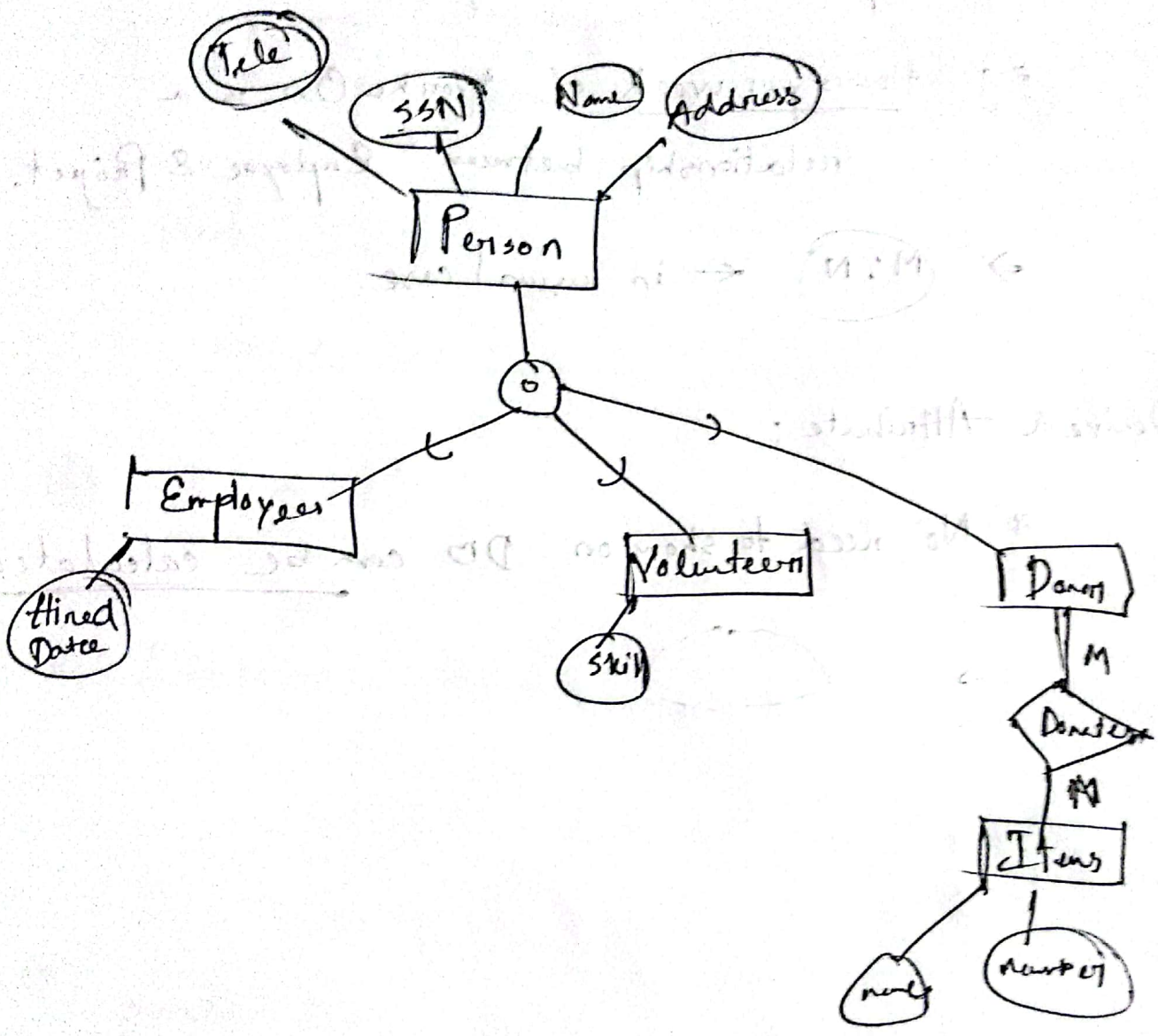
⇒ (M:N) ← in usual case

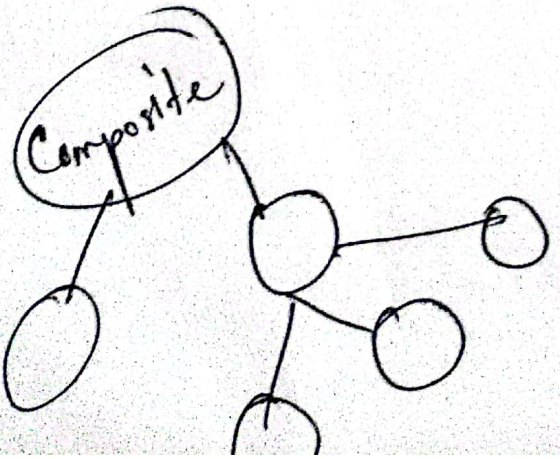
* Derived Attribute:

* No need to show on DD can be calculated.



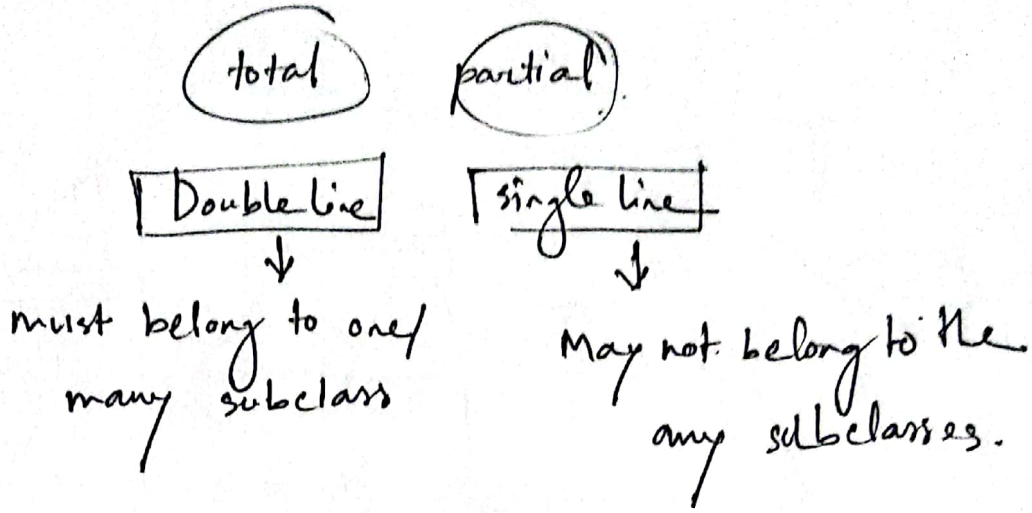
Organization

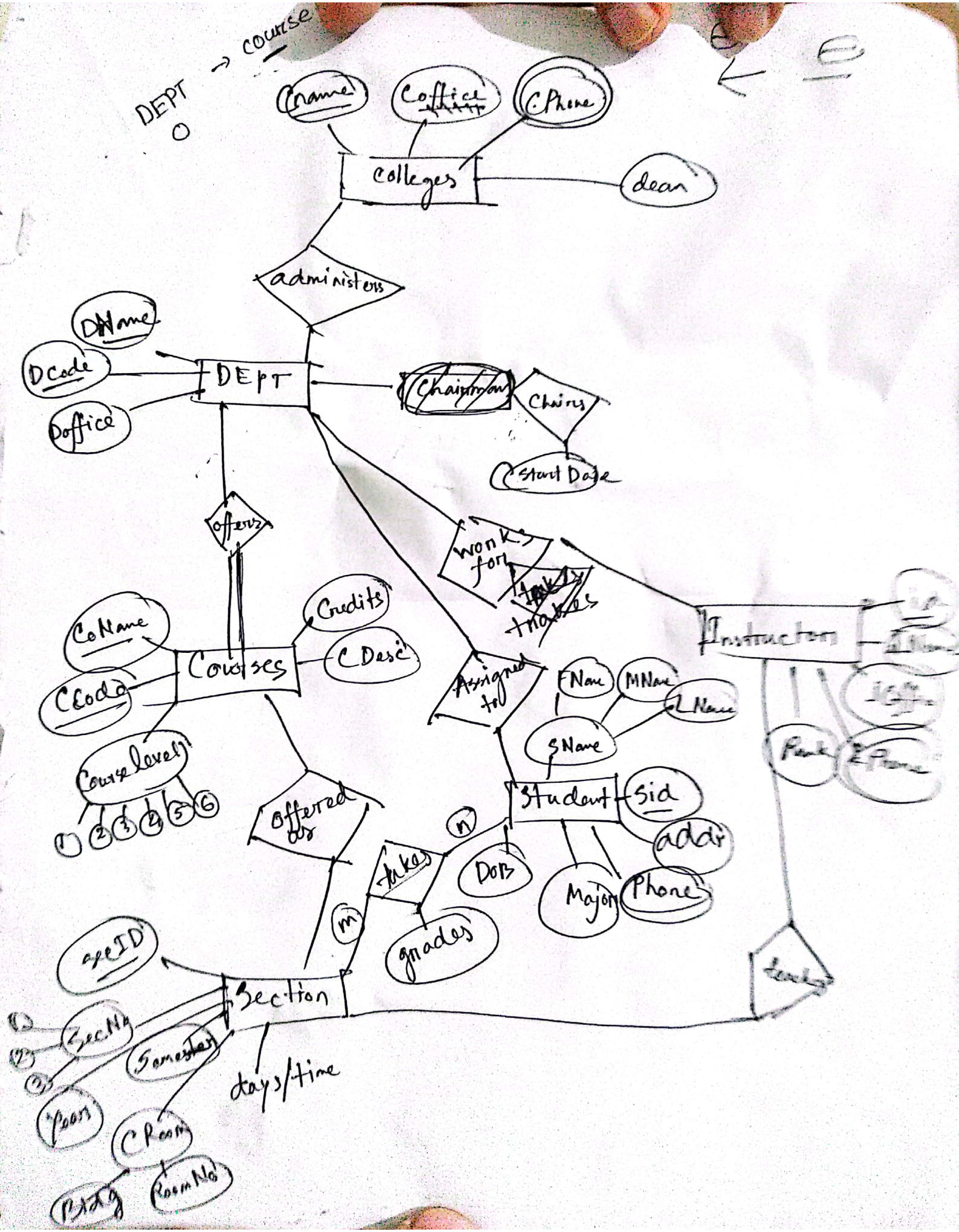


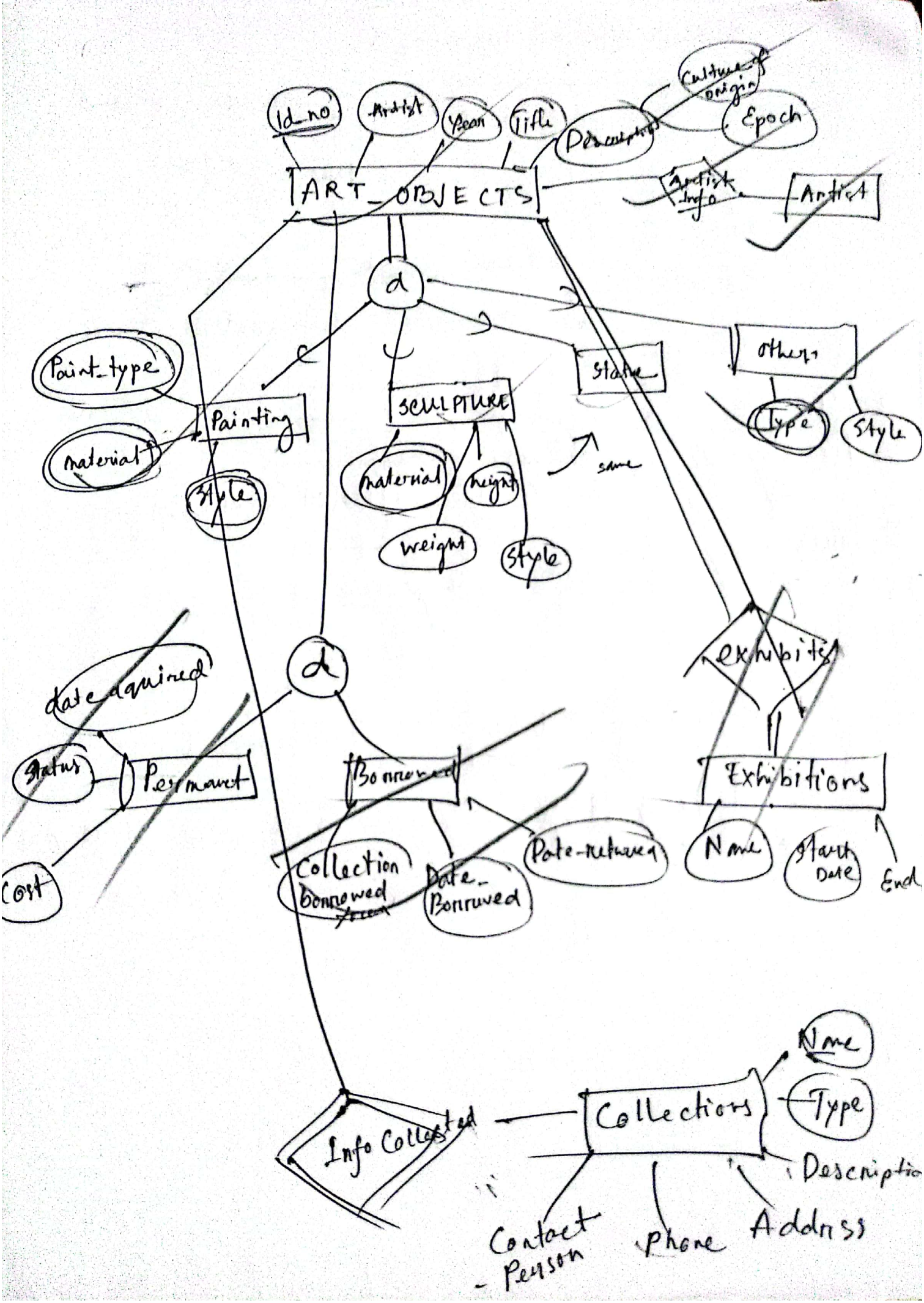


Constraints

1. Disjointness Constraints
2. Completeness Constraints




DEPT → course
Cname
Coffice
CPhone
colleges
dean
administers
DName
DCode
DOffice
DEPT
Chairman
Chairs
Start Date
offers
works for
Credits
CDesc
CName
CCode
Courses
Course level
1 2 3 4 5 6
offered as
Assigned to
FName
MName
LName
SName
Student
Sid
addr
DOB
Major
Phone
takes
m
n
grades
Instructor
Rank
EPhone
secID
SecNo
Semester
days/time
Room
CRoom
RoomNo
Bldg


ART_OBJECTS
Id_no
Artist
Year
Title
Description
Culture of origin
Epoch
Artist info
Artist
d
Paint_type
Painting
material
Style
SCULPTURE
material
weight
weight
Style
same
State
others
Type
Style
d
Date acquired
Status
Permanent
Cost
Borrowed
Collection Borrowed
Date_Borrowed
Date-returned
Exhibits
Exhibitions
Name
Start Date
End
Info Collected
Collections
Name
Type
Description
Contact - Person
Phone
Address

Refferential Integrity Constaint

10.18.23

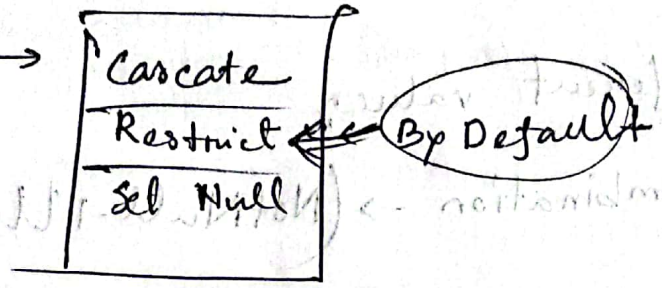
PK ← primary key
FK ← foreign key

Entity Integrity

t(PK) ≠ null

- Refferencing Relation ← FK
- Refferenced Relation ← PK

Delete →



Constraints :

- Key
- Entity
- Referential
- Domain

1. Unique → different values
2. Primary key → combination → (NotNull + Unique)
3. Foreign key → { primary key domain & not null }

Referencing Relation

↳ যে table এ foreign key

Referenced Relation

↳ যে table থেকে value নেয়া হয়

Chapter 05

informal	formal
table	relation
column header	attribute
possible column values	Domain
Row	Tuple

Table Defination → Schema of a Relationship

Populated table → State of the Relation

⑯ Relation state → subset of the cartesian product of domains of its attributes.

dom(cust_name) is varchar(25)

(A) Relational Integrity Constraints

→ conditions that must hold

1. key Constraint
2. Entity Constraint
3. Referential Integrity constraint

(A) domain constraint (implicit)

every value in tuple must be from domain of its attribute (could be 'null' is allowed)

(1) Key Constraints:

- Super key
- key
- Candidate keys
  - ↳ primary keys

unique

→ No two tuples in a relationship must have the same value for key attributes.

② Entity Integrity Constraints:

↳ the primary key attributes PK of each relation schema R is S can not have null values in any tuple of r(R).

$$\therefore t[PK] \neq null \rightarrow t \text{ in } r(R)$$

③ Referential Integrity Constrains: (Foreign key Constraint)

↳ referential integrity :- constraint used to specify a relationship among tuples in two relation

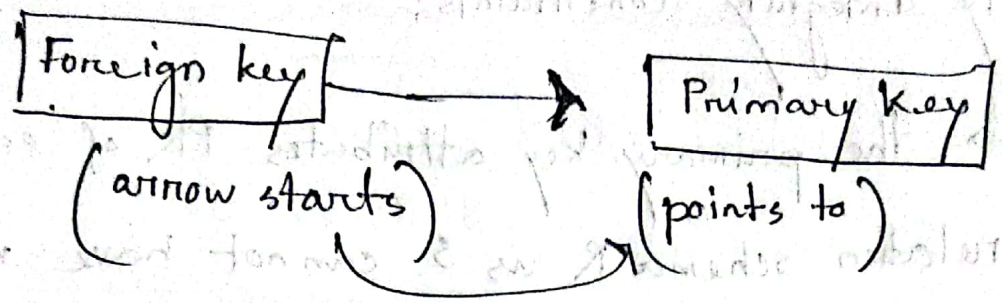
e.g. tuple t1 in R1 is said to reference a tuple t2 in R2 if $t1[FK] = t2[RK]$

↓ the value of the foreignkey (FK) of Referencing relation R1 can be →

① a value of existing primary key value of corresponding PK in referenced relation R2.

② a null. (if not part of own primary key)

Remember:



⍟ Semantic Integrity Constraints:

↳ based on application semantics

↳ can't be expressed by model

↳ SQL-99

⍟ populated DB state:

if database changes → state changes

- delete
- modify
- ~~update~~ insert

Violations

* Insert:

violates

data type violation

i) Domain ⇒ if attribute values provided for new tuple is not of specified attribute domain.

ii) key → if value of key attribute in new tuple already exists in another tuple in relation

iii) Entity → if primary key value is null.

iv) Referential → if foreign key value in new tuple references a primary key value that does not exist in the referenced relation.

* Delete:

i) Referential Integrity Constraints.

- Restrict → reject deletion
- Cascade → delete tuples that references the tuple that beig deleted.
- Set Null → set FK of referencing tuples to Null.

if not PK

(*) Update: (Modify)

may → i) Domain constraint
ii) Set Null constraint

(iii) Updating the PK

- similar to delete followed by an Insert.
- specify similar options as delete

(iv) Updating the FK

may → • violets referential integrity.

(v) Updating ordinary attribute (not PK/FK):

- can only violate Domain Constraint