Relational Calculus

- Relational calculus expression specifies what data we want to retrieve. It doesn't mention any opern. or how to retrieve (NON-PROCEDURAL)
- In relational algebra exprn. → seq. of opern (ordering of opern). (PROCEDURAL)
- whatever we retrieve using basic relational algebra operns can also be expressed using relational calculus & vice versa
  → power of expr. is same
  → A relational query lang. L is relationally complete if any relational calculus expr. can be expressed in L

Tuple Relational Calculus

Tuple variable ⇒

⇓

Range relation
+ can hold a tuple of its range relation

$\{t \mid \text{CONDITION}(t)\}$

$\{t \mid \text{EMP}(t)\}$, $t \in \text{EMP}$ → all tuples of EMP

DEPT(DCODE, DNAME)

EMP(ECODE, ENAME, BASIC, DCODE)

$\{t \mid \text{EMP}(t) \text{ AND } t.\text{BASIC} > 50000\}$

* To use condn. on tuples of multiple relns. EMP & DEPT
  — $\{t.\text{ENAME}, d.\text{DNAME} \mid \text{EMP}(t) \text{ AND } \text{DEPT}(d) \text{ AND } t.\text{DCODE} = d.\text{DCODE}\}$

Free tuple variables (variable appearing to the left of bar)

* any variable introduced in the right of the bar must be bound by a quantifier
  — Existential ⇒ there exists
  — Universal ⇒ For all

To get all tuples of EMP for which DNAME = "XYZ"
- add in cond. mapping to DEPT reln. must exist

$\{t.\text{ENAME} \mid \text{EMP}(t) \text{ AND } (\exists d)(\text{DEPT}(d) \text{ AND } d.\text{DNAME} = \text{"XYZ"} \text{ AND } t.\text{DCODE} = d.\text{DCODE})\}$

STUDENT(ROLL, NAME, ...)
SUBJECT(SCODE, SNAME, ...)
RESULT(ROLL, SCODE, SCORE)

Name of the students who have scored ≥ 50 in all subjects

→ for all subjects there must exist entry in RESULT with same SCODE

$\{ st.NAME \mid STUDENT(st) \ (\forall su)\ (NOT\ SUBJECT(su)\ OR\ (RESULT(\ldots$

making forcibly true for all non-SUBJECT tuples
i.e. if non-subject tuple then further condns not checked but condn. satisfied
→ if if was AND instead then it would be false for for non-SUBJECT tuples, therefore ∀ would fail

AND
st.roll = r.roll
AND
su.scode = r.scode
AND
r.score > 50) }

$(\forall x)(P(x))$ EQUAL TO $NOT\ (\exists x)(NOT\ P(x))$
$(\equiv)$

$(\exists x)(P(x)) \equiv NOT\ (NOT\ P(x))$

NOT $(\exists x)(P(x))$ → NOT $(\forall x)(P(x))$

no tuple should satisfy condn.

fails if atleast one doesn't satisfy

$\{t \mid NOT\ EMP(t)\}$ → all tuples from universal reln. $\notin$ EMP → infinite no. of tuples → all reln. calculus returning non-finite no. of tuples are called UNSAFE

Entity-Relation (ER) Model

→ to capture the data requirements ⇒ conceptual level for an appln.

→ ER diagram to visually represent entities & their association/relations & attrs. → to ensure handshake agreement with client

- **Entity**: an entity is a distinguishable real life object or abstract concept
  An entity has some attrs.
- collection of similar entities (same attrs)
  ⇓
  Entity set
  struc. of an entity set is defined by the entity type
- Entity type: schema/ intension of a reln.
  Name of the entity set & attrs.

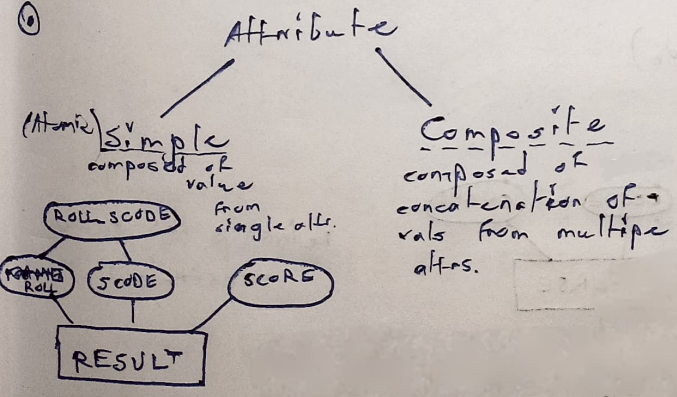
ER Model → Reln. Assocn.
⇓
b/w the entities
set of similar assocn.
⇒ relnship set
⇓
struc. defined by relationship type (what entity types are involved)

Entity types → $E_1, E_2, \ldots E_n$

Relationship set = $\{(e_1, e_2, \ldots e_n) \mid e_i \in E_i\}$

no. of entity types involved ⇒ deg of reln. in ER diag.

①



⊙ Single-Valued & Multivalued Attr.

- For each entity in an entity set, attr. has single value ⇒ SINGLE VALUED ATTR
- multiple values ⇒ MULTI-VALUED

⊙ Stored vs. Derived Attr.

- Stored: an attr. whose value can be obtained only by storing it
- Derived: the value of an attr. can be obtained from other stored attr.
  constitutes redundancy → may result in inconsistency
  eg. DOB is modified ⇒ AGE needs to be changed too

⊙ Complex Attr.

Nesting of composite & multivalued attr.

{CONTACT(ADDRESS, {PH_NO})}

{CONTACT(ADDRESS(HOUSE, STREET, CITY, PIN), {PH_NO(COUNTRY, ZONE)}

⊙ Domain/Value set

Attr. can take value from its domain or null

An attr. A_n of entity type E has a value set V

A can be defined as a func. from E to power set of V

$A : E \rightarrow P(V)$

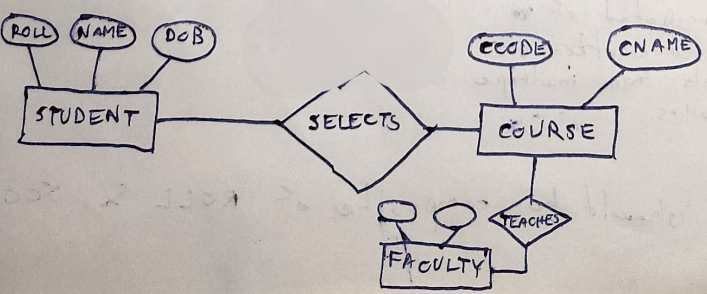
(set of all subsets of V)

If A ⇒ $A_1, A_2 \ldots A_n$

(composite attr.) V_1 V_2 V_n

V (value set) ⇒ $P(V_1) \times P(V_2) \times \ldots \times P(V_n)$

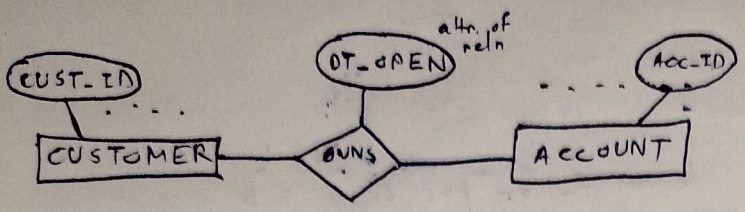
#.



15/01/26

ER Model (SKS)

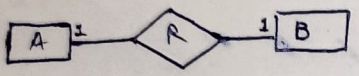
- **Constraint on Relationship**
  - reln can also have attrn.



- **Constraint** → mapping cardinality of binary relationship
  An entity can have association with how many instances of another entity type

→ **One-to-One**

An entity can take part in association with atmost one of the related entity set & vice versa



(not preferred)

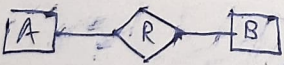
→ **One-to-Many**

A to B ⇒ one to many
An instance of A can get associated with many (zero or more) instances of B but an instance of B can have reln with atmost one instance of A

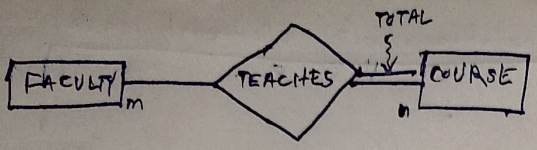
→ **Many-to-One**

Many instances of A can get associated with atmost one instance of B

→ **Many-to-Many**



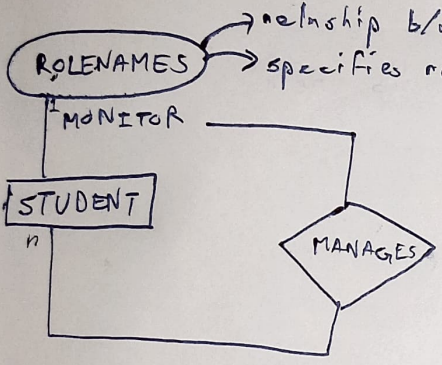
- **Participation Constraint** → whether an entity is bound to take part in an association (minimal cardinality of the reln.)
  - **Partial participation**: instances can exist in entity without assn. with other entity's instance
    An entity partially participating in a reln means its instance may not have assn. with instance of other entity
  - **Total participation**: an entity totally participating ⇒ an instance must get related to instance of other entity type



Faculties can exist without teaching a course but every course must have Faculty association.

#In total participation, the entity totally participating as it depends on the other entity

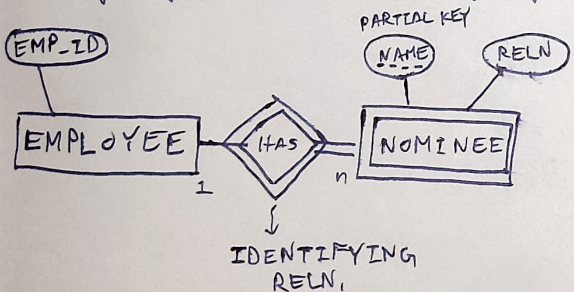
- Structural Constraint
  - Mapping Constraint
  - Participation Constraint



MONITOR is also of STUDENT entity type, it manages many STUDENT entities with ROLENAMES != MONITOR

- Weak Entity Type
  - An entity type which has no key of its own

#Weak entity type must totally participate with another entity type (owner entity type)



#Given an instance of owner entity type, by traversing the reln., one can find the subset of weak entity set related to owner instance.
In that subset the attr. maybe simple or composite. Attr. of weak entity type acting as discriminator to identify the instances of weak entity set is called partial key of weak entity type

- Strong Entity Type : entity type having its own key

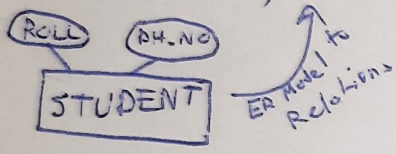
- **Mapping From ER model to relations of relational model**

1) **Strong entity type**

⇒ A reln. corresp to the strong entity type with the attr as schema

- composite attr. ⇒ replaced by its constituent simple attrs
- multi-valued attr. ⇒ create a separate reln. for such attr. Copy the key of corresp. entity type in the new reln. In new reln, copied key will be Foreign key M.V. attr. & copied key will form composite key for new reln.

STUDENT(ROLL, NAME...)
PHONELIST(ROLL, PHONE_LIST)



2) **Weak entity type**

Consider a corresponding reln. with its reln.

Schema: key of owner entity type U its own attr.

↓

FK referring to owner entity type

Key: Key of Owner U Partial key

Aggregation :- Aggregate Func.

EMP(ECODE, DCODE, ENAME, BASIC, GRADE)

$SUM_{BASIC}(EMP)$ ⇒ Gives sum of BASIC for all tuples, acts on all except null

$COUNT_{ECODE}(EMP)$ ≈ STD DEV. MAX MIN

Grouping expression, optional, if not given, all tuples will be taken

$\mathcal{F}_{SUM_{BASIC}, AVERAGE_{BASIC}}(EMP)$

DCODE

Multiple Aggregate Func.

DCODE	SUM_BASIC	AVERAGE_BASIC

EMP(ECODE, ENAME, DCODE, GRADE, BASIC)

→ For each dept, each grade, what is the avg. basic

Groups — (D1, A), (D1, B), (D2, A), (D2, B)

⇒ Grouping expr. : $\mathcal{F}_{DCODE, GRADE}(EMP)$ AVERAGE BASIC

→ $T(DCODE, TOTAL\_BASIC) \leftarrow \mathcal{F}_{DCODE\ SUM\_BASIC}(EMP)$

$T(DCODE, TOTAL\_BASIC)$

$\sigma_{TOTAL\_BASIC >= 100000}(T)$

or

$\sigma_{TOTAL\_BASIC}(\rho_{T(DCODE, TOTAL\_BASIC)}(DCODE\ \mathcal{F}_{SUM_{BASIC}}(EMP)))$

O/p schema → Grouping expression (if any) AGGREGATE

Per collection → one tuple in the o/p schema.

For each Dept ⇒ Each Grade ⇒ Average Basic