26.7.22

"Love what you do, do what you love."

DBMS

— Structs and arrays can store data but using proper data structures can reduce time complexity of CRUD operations.

— DBMS can store data in a single place and answer queries from user based on sort or a condition.

— Programming → Data Structures → DBMS → Data Analytics

storage, size of variables — efficient storage, reducing time/space — optimise access, update, space — Multiple DBs.

Database -

— Database is a collection inter-connected data items. It is represented in tabular or graph form generally.

— Each row of the database can be treated as an object with the columns as its properties.

Management -

Management refers to Access, Updation (Modify, insert, delete), Retrieval.

Data Models -

Network, Hierarchical model scrapped due to closer relation to physical view.

28.7.22

Entity - Relationship model -

It has lesser use than relationship model however it has applications in many domains.

Relational Model -
- In a relational model, all data is stored in various tables. Each table has rows and columns.
- 1 row is called a record.

Schema and Layout -
- Schema is the definition for 1 record in a table. Similar to types and variables in programming language.
- Logical Schema - logical structure of database.
- Physical schema - physical structure
- Instance - Current status of database with all it's data.

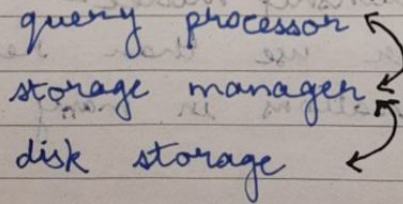
Database languages

DDL → Create db, Delete db
DML → Manipulate db, Insert, update, delete, add/r

- char(5) is fixed size of 5 characters
- varchar(20) is maximum of 20 characters
- numeric(8,2) is 8 before decimal and 2 after decimal.

- DML can be procedural (Relational algebra, PL-SQL) or declarative (TRC) Tuple Relational Calculus

2.8.22



Models → DB languages → Normalization → storage + query optimisation → Transaction agent

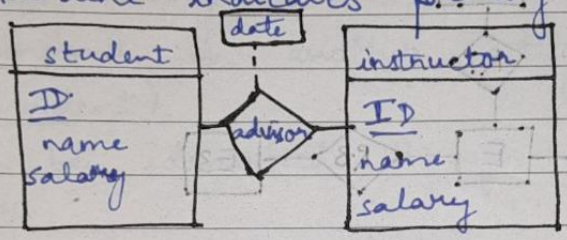
Relational model –

– K is a superkey of R if values for K are sufficient to identify a unique tuple of each possible relation r(R).
– Candidate keys are minimal superkeys.
– One from candidate is primary key.

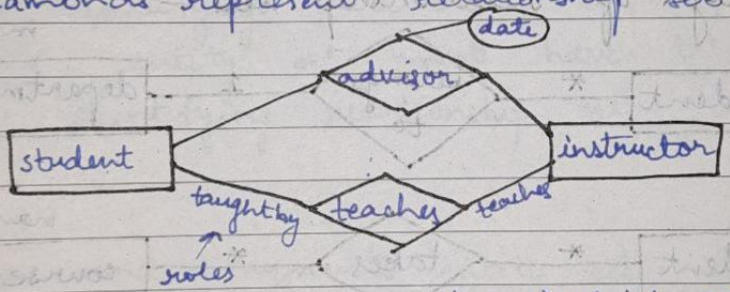
4.8.22 "Love what you do, do what you love."

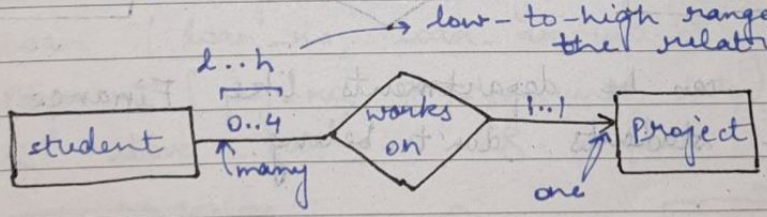
Entity –

– Rectangles represent entity set.
– Attributes are listed inside entity rectangle.
– Underline indicates primary key attribute.



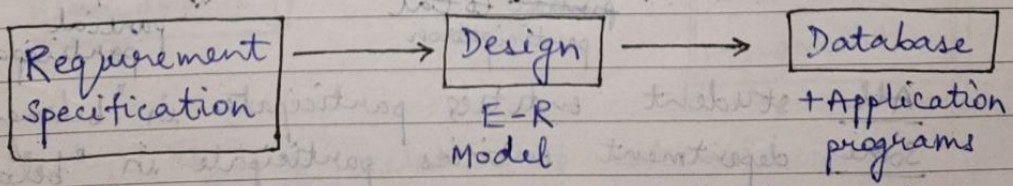
– Diamonds represent relationship sets.



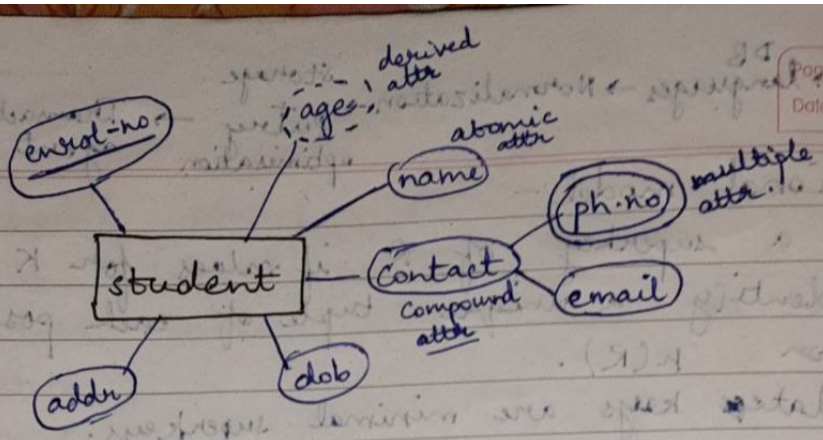


many-to-one relation

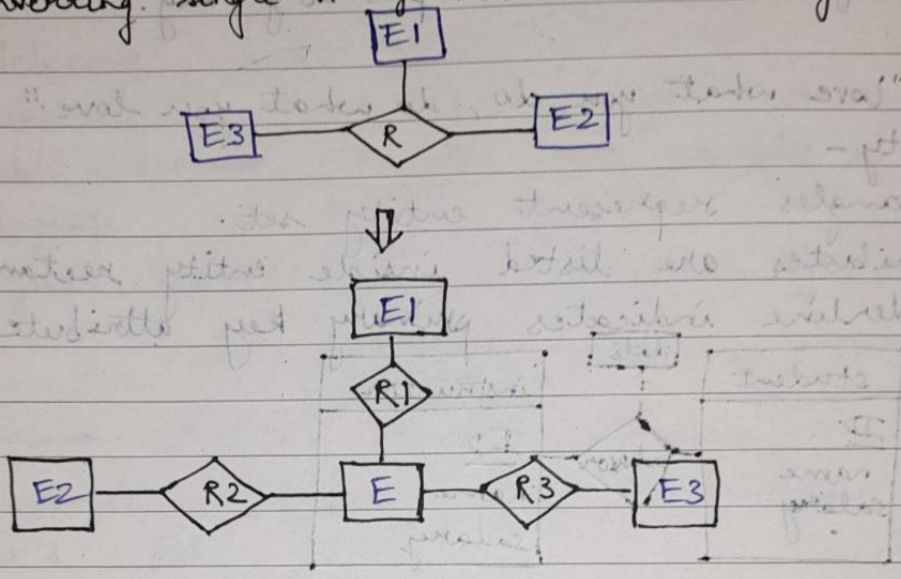
11.8.22



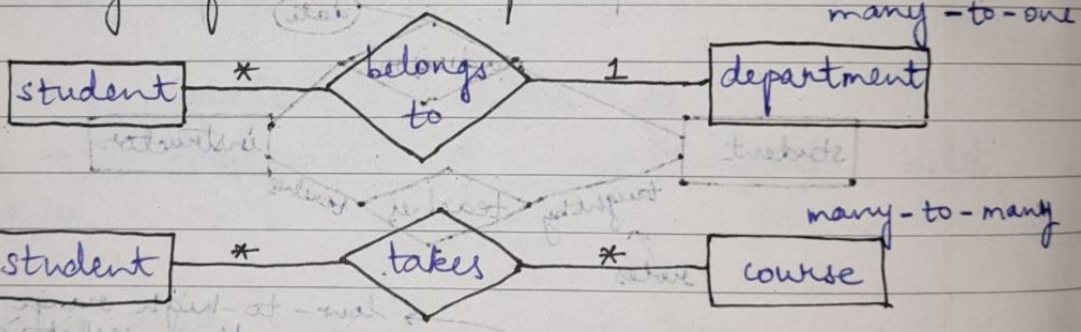
1. Entity sets
2. Attributes
3. Key constraints
4. Relationship sets
5. Cardinality



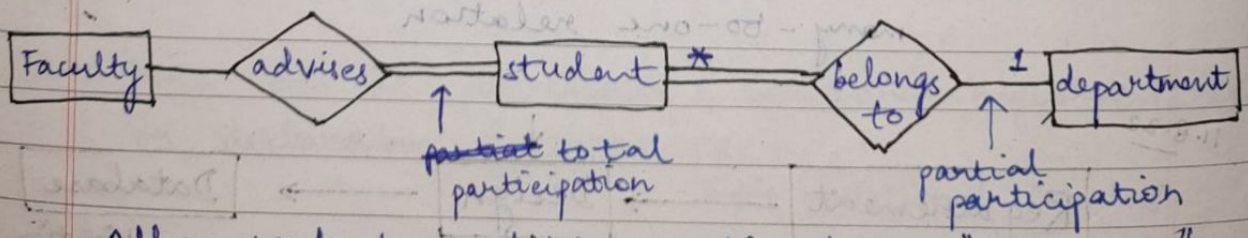
* Converting single n-ary relation to binary.



* Cardinality of relationship sets —



— There can be departments like Finance, RAC, T&P, where students don't belong.



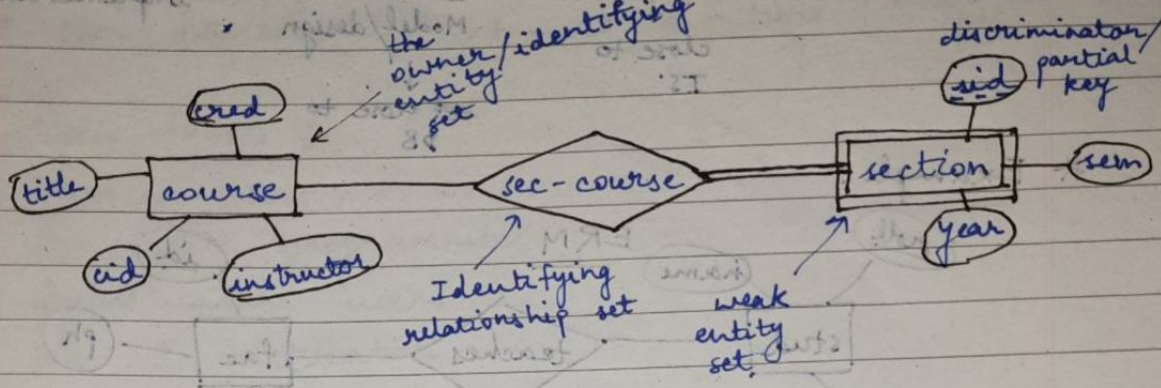
All student entities participate in "belongs to".
Some department entities participate in "belongs to".

16.8.22

"Love what you do, do what you love."

ERM to Relational schemas

- Real world → Entity-relation model → Relational schemas → DB + DBMS



* Weak Entity set -

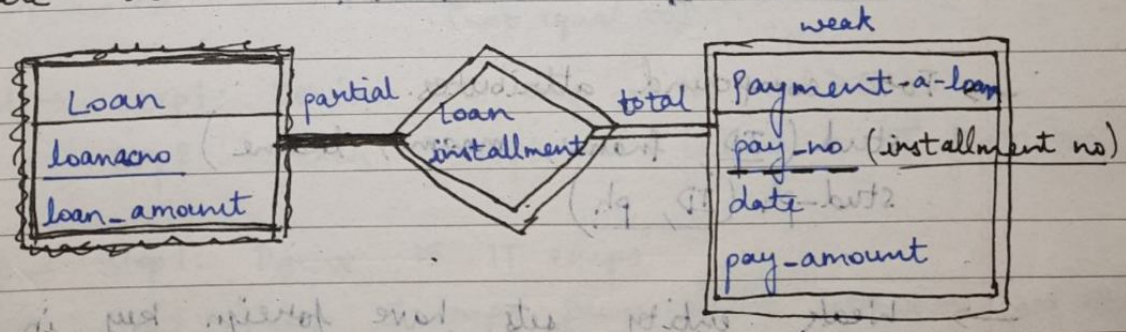
- Dependent on another entity set. (Existence dependent)
- Can be identified through uniqueness of primary key.
- It has a partial key and an owner entity set and an identifying relationship set.
- Weak entity set must have total participation in identifying relationship set.

Q - Loan and Payments entity set.

Loan (loan_no, loan_amount)

Payment (pay_no, pay_amount, date)

Relate above two in ER model.



- Primary key of relationship set → (loan_dc_no, pay_no)

- Only the primary key of "Many" side is not enough in case of weak entity set.

23.8.22

"Love what you do, do what you love."

E-R model to Relational model

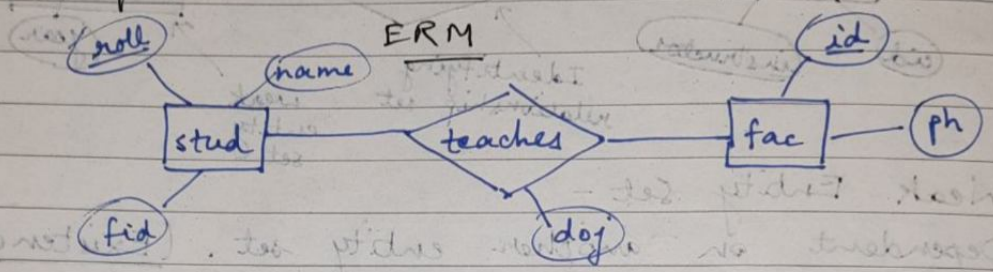
Requirement specification → E-R modelling → Relational modelling → DB + DBMS implementation

E-R modelling: ↓ close to IS

Relational modelling: Model/design ↓ close to DB

* Example:

ERM



RM

stud (roll, name) // stud (roll, name, id) ← foreign key

fac (id, ph)

teaches (roll, id, doj).

DB

stud

roll	name

faculty

id	ph

→ For compound attributes.

stud (ID, fname, mname, lname)

stud-ph (ID, ph)

→ Weak entity sets have foreign key, in general.

loan (ac, amt)

pay (ac, pay no, amt, date).

→ In case of partial participation, we can have foreign keys with NULL values.

30.8.22

"Love what you do, do what you love."

Relational Algebra

* Database languages ⇒
  - Procedural / Record-at-a-time [what data + How-to-get]
  - Non-Procedural / Set-at-a-time [What data to get]

* User levels ⇒
  - Novice users
  - Application programmers
  - Sophisticated users
  - Db administrator

1. Selection (σ)

Emp(eno, ename, dept, sal)

> Select details of all employees with salary > 50k

$\sigma_{sal>5000}(Emp)$

> Select details of all employees ~~of~~ belonging to IT dept.

$\sigma_{dept='IT'}(Emp)$

> $\sigma_{dept='IT' \wedge sal>50000}(Emp)$ (AND)

> $\sigma_{dept='IT' \vee sal>50000}(Emp)$ (OR)

→ Operators –

$=, \geq, >, <, \leq, \neq$ <> (not equal to)

M1 → step1: Derive 100 emp with sal > 50k.
step2: Fetch 5 emps who belong to IT from 100.

M2 → Step1: Derive 15 IT emps
Step2: Fetch 5 emps from above set whose sal > 50k

$\sigma_{sal>50k}(\sigma_{dept='IT'}(Emp))$

* σ is commutative.

$\sigma_{cond1}(\sigma_{cond2}(R)) = \sigma_{cond2}(\sigma_{cond1}(R))$

2. Project Π

Select specific details : Eliminates duplicate tuples.

> Select eno, ename from Emp.

$\Pi_{eno, ename}(Emp)$

> $\Pi_{eno, ename}(\sigma_{dept='IT'}(Emp))$

will not work

* Π is not commutative.

Q- Select name, dept of all employees with sal > 50k

$\Pi_{ename, dept}(\sigma_{sal>50k}(Emp))$

3. Degree of r → Number of attributes / cols

Cardinality of r → Number of rows / tuples

$t \leftarrow \Pi_{list}(r)$

$|t| = |r|$ when list = {atleast one candidate key}

3. Rename ρ

$\rho_{Employee}(Emp)$ → Rename schema

$\rho_{(enum, ename, dept, sal)}(Emp)$ → Rename attribute

$\rho_{Emp\_dst(enum, ename, dep, sal)}$ → Rename attribute & schema

→ This operation creates a new copy with the changed names.

6.9.22

"Love what you do, do what you love."

Relational Algebra

Q- Find the roll number of topper.

$$\Pi_{roll\text{-}no}(student) - \Pi_{t.rollno}\left(\sigma_{student.marks > t.marks}\left(student \times \rho_t(student)\right)\right)$$

Q- Emp (eno, ename, type)
Course (code, eno, cname, nstudents)
student (rollno, name, code)
Find all students (roll, name) studying in course(s) conducted by Prof. RN.

$$\Pi_{rollno, sname}\left(\sigma_{ename='RN'}(Emp \bowtie student \bowtie course)\right)$$

- Theta Join $\bowtie_\theta$ COMMUTATIVE

$$r \bowtie_\theta s = \sigma_\theta(r \times s)$$

$\theta = <, \leq, >, \geq, =$

Assignment Operator $\leftarrow$

$$r \leftarrow \Pi_{t.rollno}\left(\sigma_{student.marks > t.marks}\left(student \times \rho_t(student)\right)\right)$$

$$result = \Pi_{rollno}(student) - r$$

Division $\div$

Q- Emp (eno, ename)
Workson (eno, pno)
Project (pno, pname, dur)

List eno of employees who work in ALL projects, same as 'Smith'.

t ← pno associated with 'SMITH'.

$$\Pi_{pno}\left(\sigma_{ename='SMITH'}(Emp \bowtie_{Emp.eno = Workson.eno} Workson)\right)$$

$t1 \leftarrow$ All (eno, pno) pairs σ (Workson)

result $\leftarrow t1 \div t$

Q- Branch (br_name, city)
Customer (cid, acno)
Account (acno, br_name)
Find customers who have an a/c at ALL branches located in 'Shibpur'.

$t1 \leftarrow$ all br_name in 'Shibpur'.
$t2 \leftarrow$ (cid, br_name) pairs
$t2 \div t1$

$t1 \leftarrow \Pi_{br\_name}(\sigma_{city='Shibpur'}(Branch))$
$t2 \leftarrow \Pi_{cid,\ brname}(Customer \bowtie Account)$

Extended RA operators –

1. Left Outer Join $\overset{\Rightarrow}{\bowtie}$

r			s		$q \leftarrow r \overset{\Rightarrow}{\bowtie} s$			
a_1	a_2	a_3	a_1	b_1	a_1	a_2	a_3	b_1
1	X	Y	1	P	1	X	Y	P
2	X	Z	2	Q	2	X	Z	Q
3	W	Z	4	R	3	W	Z	NULL

Unmatched values from right relation padded with NULL.

2. Right Outer Join $\overset{\Leftarrow}{\bowtie}$

r			s		$q \leftarrow r \overset{\Leftarrow}{\bowtie} s$			
a_1	a_2	a_3	a_1	b_1	a_1	a_2	a_3	b_1
1	X	Y	1	P	1	X	Y	P
2	X	Z	2	Q	2	X	Z	Q
3	W	Z	4	R	4	Null	Null	R

Unmatched values from left relation padded with NULL.

3- Full Outer Join ⟗

r

a_1	a_2	a_3
1	X	Y
2	X	Z
3	W	Z

s

a_1	b_1
1	P
2	Q
4	R

$q \leftarrow r \text{ ⟗ } s$

a_1	a_2	a_3	b_1
1	X	Y	P
2	X	Z	Q
3	W	Z	NULL
4	NULL	NULL	R

8.9.22

Aggregate Function –

1. Group-by $\mathcal{G}$

$\mathcal{G}_{\text{count(eno)}}(\text{Emp})$ → Distinct enos appearing in Emp table.

$\mathcal{G}_{\text{max(marks)}}(\text{student})$; where dID is grouping attribute

count()
max()
min()
average()
sum()

functions

2. Generalised Projection

$\Pi_{\text{roll, marks} * 0.5}(\text{student})$

Insert, Update, Delete

I student ← student ∪ {10, '...', 98, ...}

student ← student ∪ student1

(should have same columns and datatype)

D student ← student − $\sigma_{\text{dID} = \text{'CST'}}$ (student)

U student ← $\Pi_{\text{roll, name, dept, marks} * 1.2}$ (student)

DB Languages –

Procedural – RA, PL-SQL

Non-procedural – TRC, SQL

8.9.22

"Love what you do, do what you love."

Tuple Relational Calculus

$\{t \mid t \in Emp \ (t[sal] > 50k)\}$

Variable — Predicate

$\Rightarrow \{t \mid P(t)\}$ — $P(t)$ is predicate

* Forms of $P(t)$ —
  - $t \in R$
  - $t[A] \ \theta \ c$ — $\theta \in \{=, <, >, \leq, \geq\}$; c : constant
  - $t_1[A] \ \theta \ t_2[A]$

$\{t \mid t \in Emp \ \wedge \ t[sal] > 50K\}$

$\{t \mid t \in Emp \ \wedge \ t.sal > 50K\}$