25/7/18

Collection of data and programs is DBMS.

Instruction + platforms + Storage → Cloud.

File system problem

→ Storage problem
→ File format
→ Concurrency
→ Security

* How do we retrive data when data is lost from the server?

LDAP → used in single signo

Applications of DBMS

→ Banking, Airlines, Institutes, Website, Recovery Single sign on, etc.

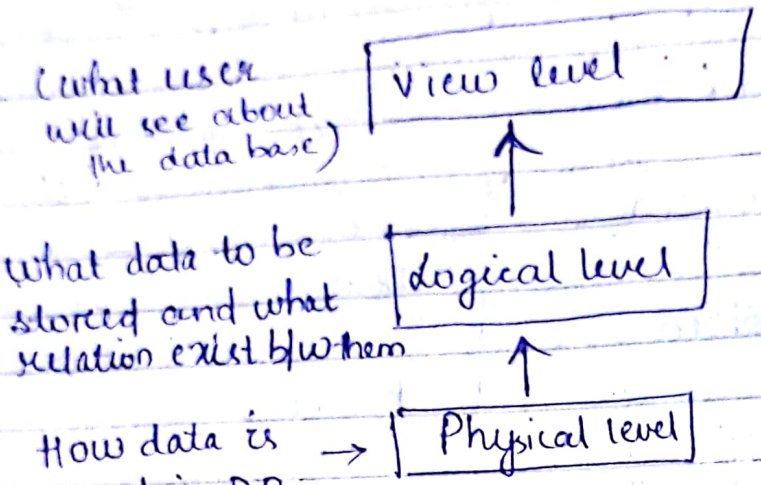
Relation DBMS → ~~table~~ data is stored in form of ER diagram (entities relation diagram) table.
↳ to understand DBMS to laymann.

↳ DBMS → SQL is an e.g.,

* Because of the file system problem, we moved to DBMS.

* Data:- collection of raw facts & figures.

Scanned by CamScanner



Data independence :- when we make change in one level without affecting another level.

Eg:- Changes in LL without changing PL → Physical independence

Changes in VL without changing LL → Logical independence.

→ At a particular moment, Home what is the description of the database → instance.

→ schema :- Overall description of the database at any moment / time.

↳ description includes structure, constraint, data types used, etc.

Scanned by CamScanner

Constraints like domain constraints, consistency constraints, etc.

Types of data model
ER data model, record data model, object oriented data model, semi structured data model.

Unstructured data is big data (known as) **

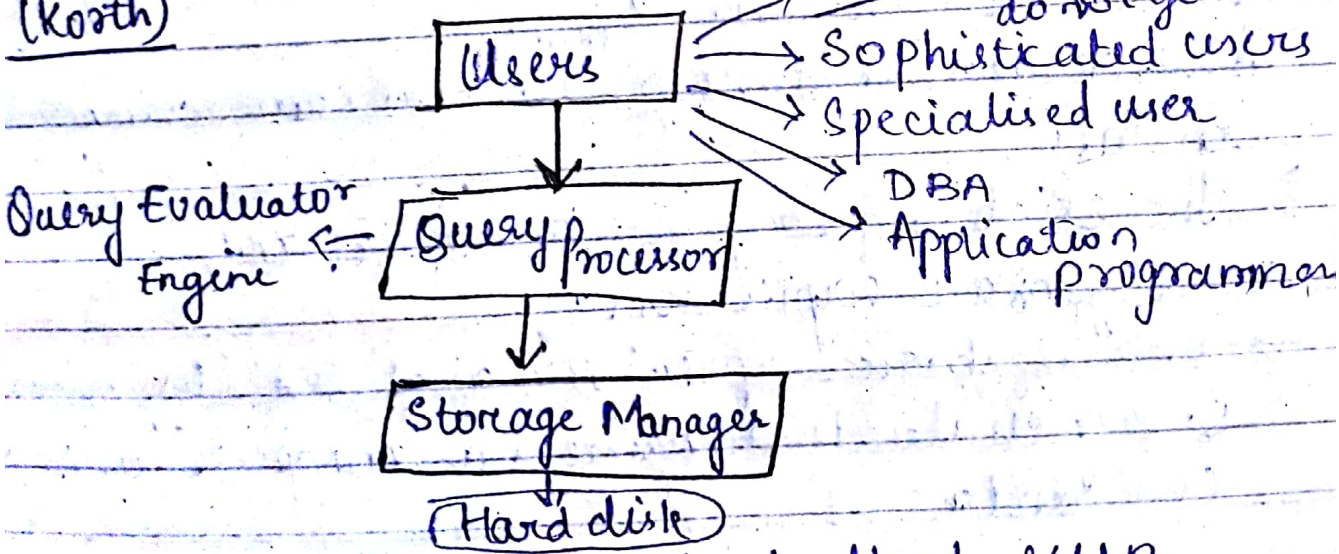
↳ No sql data bases.

↳ Data of any type (not a particular structure) (can be photos, texts, audio, etc).

26/7/18

DBMS Architecture.

(Korth)



They are the data analyst that run the queries on the database - Sophisticated users App. programmers - They are the programmers that generate interface.

Scanned by CamScanner

Those create specialised Database (non-traditional database) — Specalised users

DBA: Database administrator → that has central command over programs and data both.

→ When to login, where to keep, check points and overall maintenance of DB is handle by DBA.

Query Processor.

→ Used to process the query

→ It contains compilers and interpreters.

→ Before this, they have DDL and DML compiler/interpreter.

→ Data Definition language :- It is defining the overall schema. truncate

→ Alter, create, drop, etc are some commands in this.

→ The english query is converted into algebraic expression.

→ Then algebraic expression is analysed (like BODMAS)

→ Query evaluation engine has complete access over data.

Q Why this architecture is called as two tier architecture?

Q How DB works in client server application?

Scanned by CamScanner

Data Base Languages

① DBL → defines the schema of the DB or the overall structure of DB
Create, alter, drop and truncate → commands.

② DML → Data manipulation lang.
Select, insert, update and delete.
↳ commands

③ DCL → Data control lang.
Used to grant & revoke permissions.
Grant & Revoke → commands.

④ TCL → Transaction control language.
Commit and roll back → commands.

Data models:

① Relational data models → Here we deal data as tables.

Eg:-

Roll No.	Name	Addrss.	Phn. No.	Grades
1	X			AB
2	Y			BC
3	Z			C
4	A			D
5	B			A
6	Q			B
7	R			C

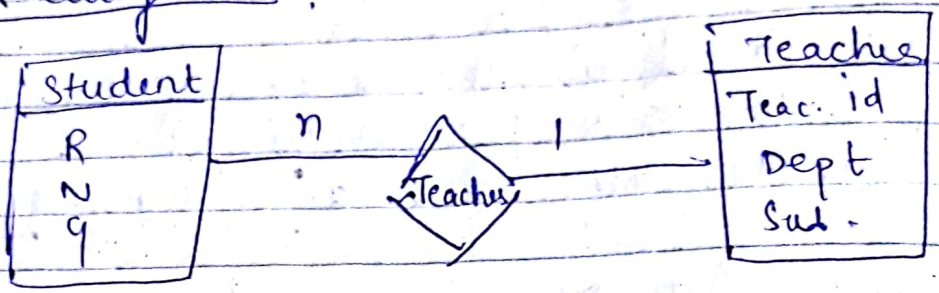
Primary Key → Roll No. (which is always unique)

→ attributes.

→ 1 record or 1 tuple

Scanned by CamScanner

② ER diagram :



③ Object oriented model :

→ R Data models Extension to deal with complex data types.
→ Here we include the concepts of OOPs.

④ Semi-structure data model

→ Basically used for websites
→ If we include XML features in RDBMS it becomes semi-structure data model

Network and Hierarchical data models are used before RDBMS.
→ In N/w we use graph data structure
→ In Hierarchical we use tree data structure

Scanned by CamScanner

DML

Procedural (What data to be used and how to retrieve that data).

Non-procedural (sql) (we have to specify only what data to be used and not that what things we have to use and how to retrieve data).

Constraints:

① Domain → This attribute cannot be beyond the domain.
→ ie, we define certain domain for the data.

② Referential integrity → we using attribute of one table in another.

Dept

Depid	Name
1	CSE
2	Maths
3	Mech.

OpenElective

CId	CName	DId
1	X	1
2	Y	2
3	Z	2
4	KL	1

Here we are using one key primary key in another (like dept. key is used in open elective)

Scanned by CamScanner

(3) Assertions: → We keep certain assertions (conditions that has to be true always).

(4) Authorization:- Permissions granting that which users can perform which commands.
like read permission granted user cannot update data, if he tries to update then the query will not be executed.

Q Create a Referential integrity data base

Students - Database - Academic

RollNo	Name	Branch	Year	Phone No.	Grades
1	Muskn	CSE DD	3rd	123	A
2	Keshu	Mangment	1st	456	A
3	Kalo	Commerce	5th	789	AB
4	Shiv	Commerce	3rd	012	B
5	Ashu	CSE DD	4th	001	B
6	Pro	CSEDD	4th	101	AB
7	Archu	ECE	3rd	160	BC

Scanned by CamScanner

Student – Curricular.

→ Reference to the previous table.

Activity id	Name	Std. RollNo **	Level
1	Dance	1	Beginner
2	Singing	6	Basic
3	Sports	2	Intermediate

27/7/18

ER Diagrams.

→ Conceptual schema → logical structure.
→ ~~Too~~ ER Diag. represent conceptual schema.

E.R. Diagrams

- Entity
- Attribute
- Relationship

→ Problem of redundancy (we have keep care of this problem).
→ Incomplete information.
(Suppose we have 2 tables → courses offered and all courses) then if there is no table of all courses, then we simply can't add subject to course offered if that course is not offered that time

Entity → a real world thing having distinguishable features

→ Attributes are the property that are related to the entity or that describe the entity.

→ Relationships exist between two entities.

Entity set is a set of entities that have similar property.

→ Entity set is never disjoint set.

Simple attributes → which cannot be further divided. eg; gender.

Composite attributes → which can be further divided. e.g. Name, address.

Single valued → that has one value. eg. Name.

Multi valued attributes → that has more than one value. e.g; address, phone no.

Stored and derived attributes

→ can be derived from the database (age if we have dob)

↳ simply stored in the database

→ Attributes can have null values. (either the value is missing or unknown)

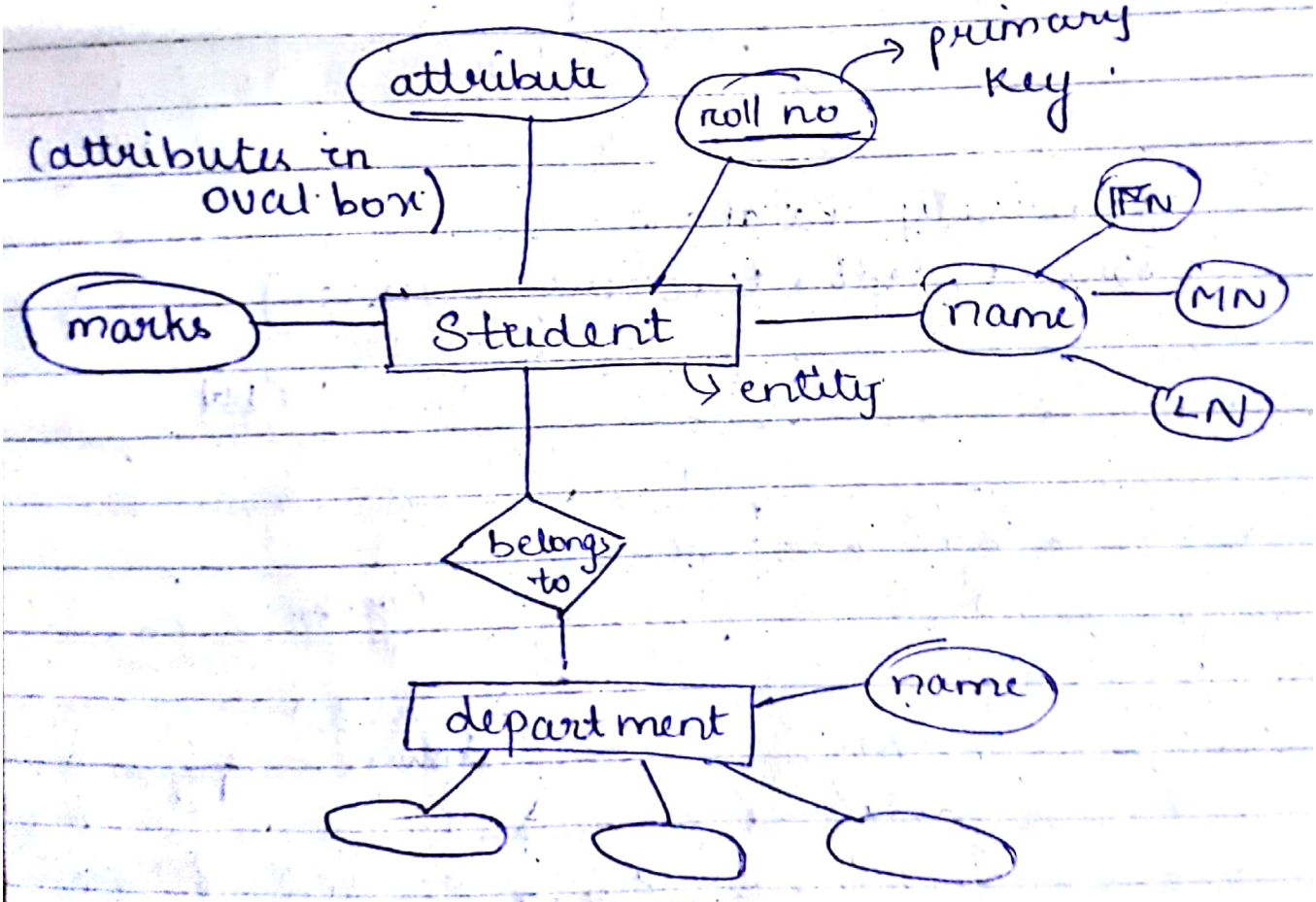
↓ value exist but we don't know it.

↳ we don't k

Scanned by CamScanner

Relationship is an association between two entity.
Eg, Manager and department, so relationship exist is manages.

Descriptive attributes → that describe the relationship b/w two entity.
Eg, joining date of manager.
Project carried out by manager.

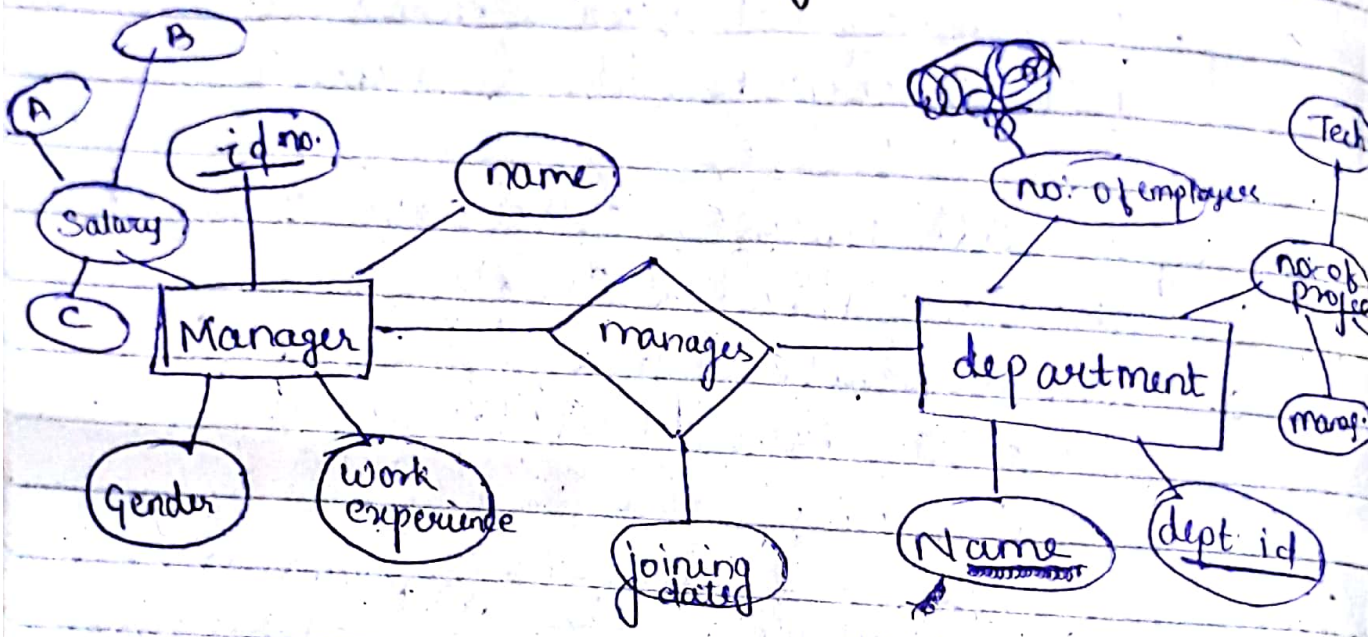


ER diagram

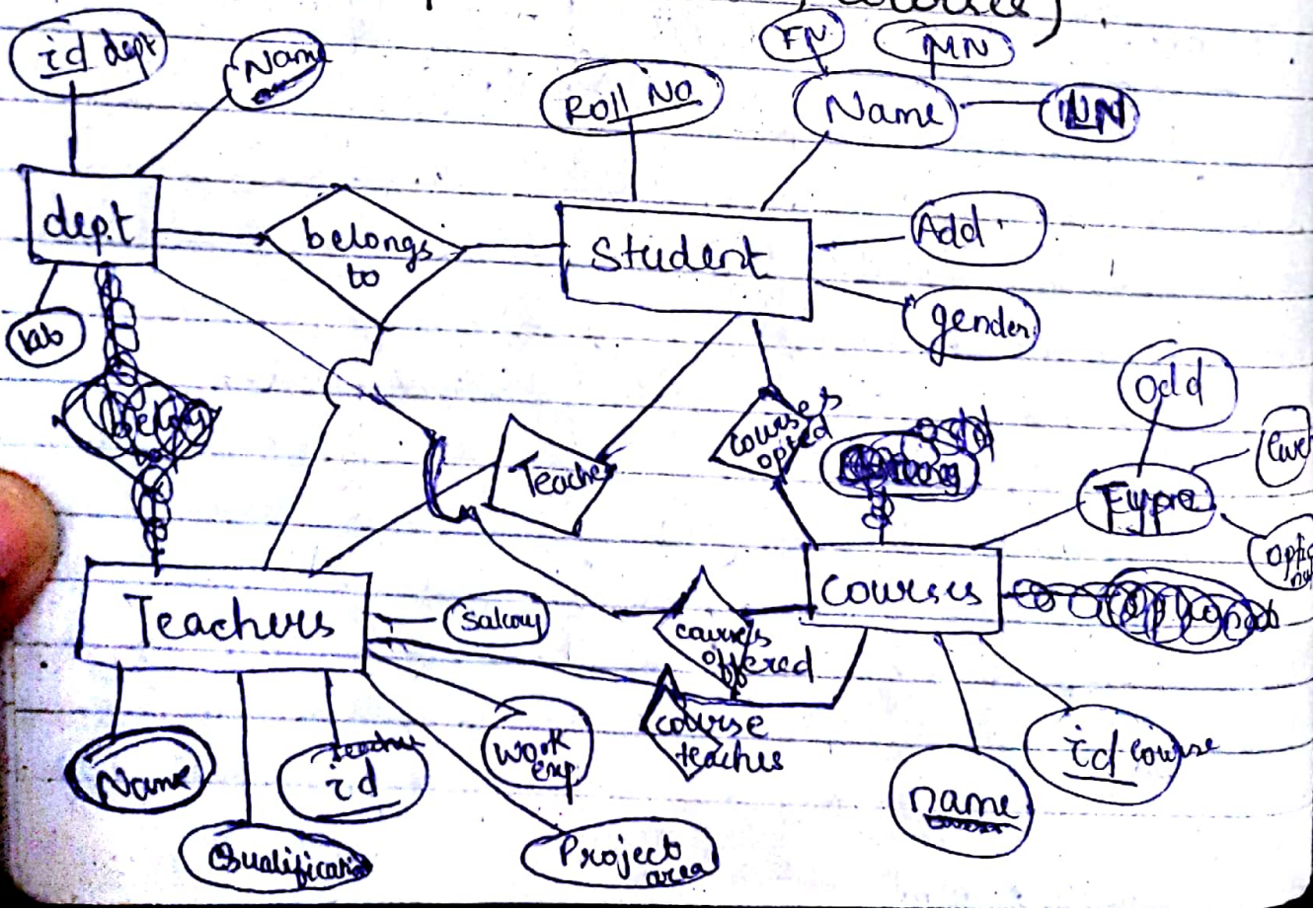
know whether the value exist or not.
(e.g, middlename)

Scanned by CamScanner

Q Manager and department
(Relationship manages)



Q University database.
(Student, dept, teachers, courses)



Scanned by CamScanner

8 Airline booking portal.

01/8/18.

Relationship Set (RS)

Political

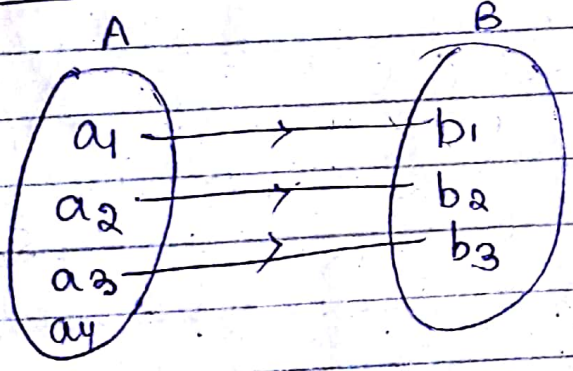
Set of relations having similar properties.

Enter disp

→ Entity are more in no!- than relations so we can say RS C Entity set.

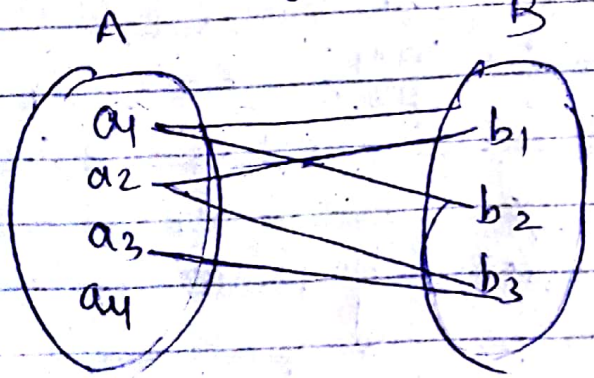
Relationship Cardinality

(1) One-to One cardinality



eg:- Name & Student Roll No.

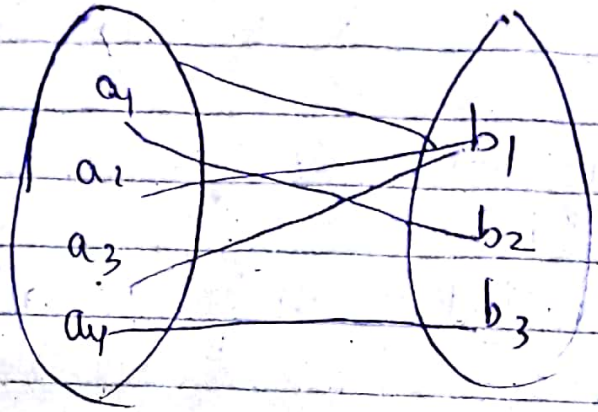
(2) One-to many Cardinality



eg; Student & Subject opted.

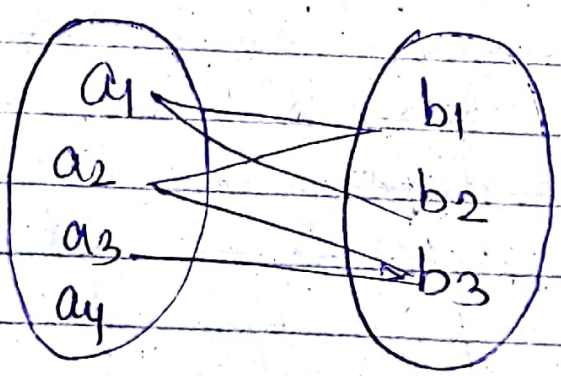
Scanned by CamScanner

(c) Many to One.

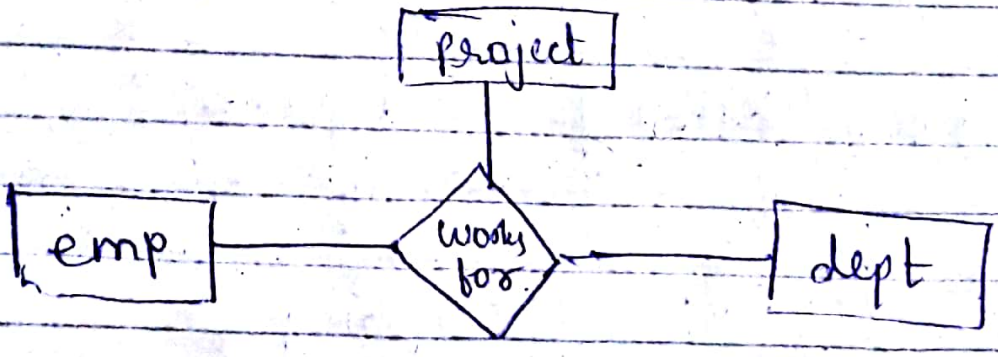


eg; Student & Teacher.

(d) Many to Many.

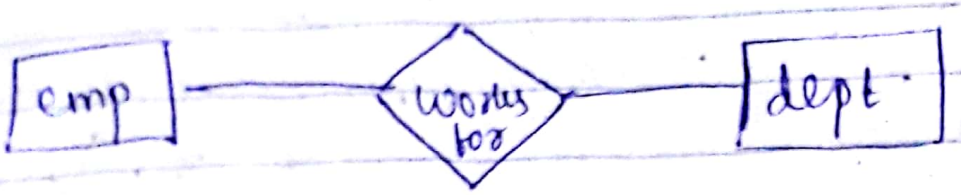


eg; Many students can opt many subjects.



Ternary relation.

Scanned by CamScanner



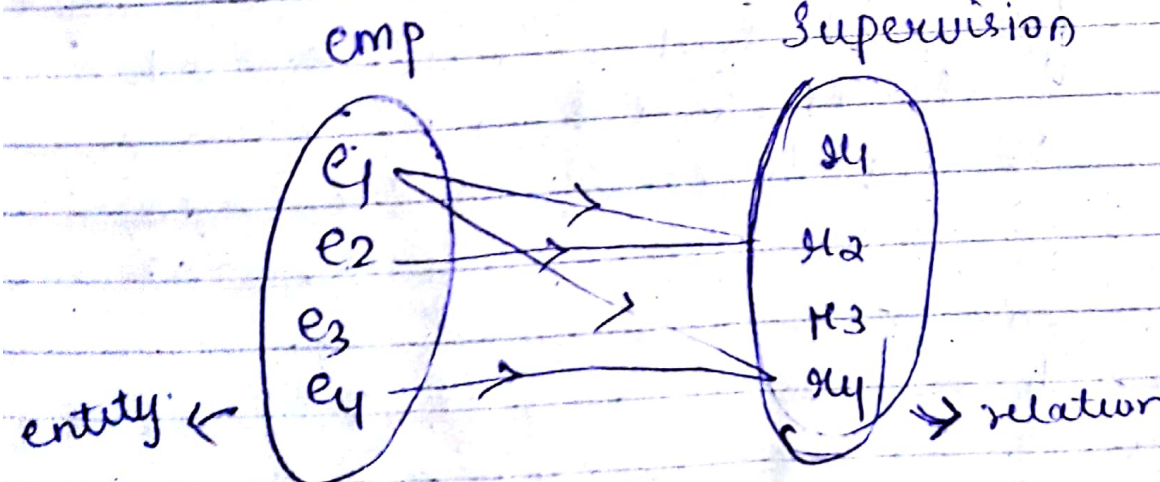
Binary relation.

Contraints.

① **Role Name** (Recursive relationship).

One entity can play many roles or have many relation.

eg:, Manager manges a dept which is an entity but he is also an employee (which is also an entity).



Employee e_1 manages (r_2) e_2 and manages (r_4) e_4.

Relation is also known as participation.

Scanned by CamScanner

Total participation

Eg → Student after enrolment can participate in a class.

Partial participation.

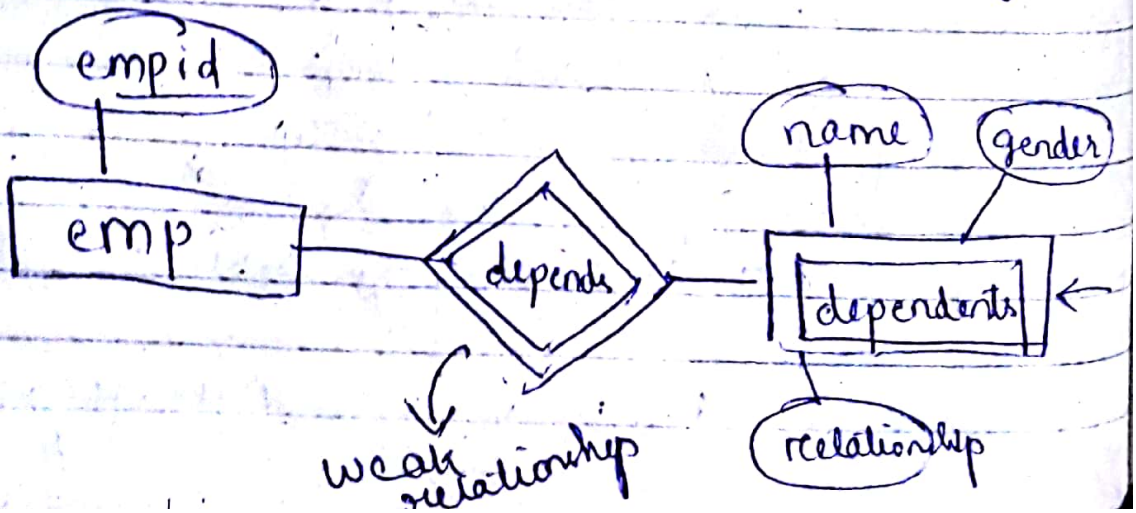
Eg: → Every employee can't be a manager.

→ If for any entity will have primary key then it is a strong entity.

→ For any entity if we don't have primary key then it is a weak entity.

Eg; If we don't have roll number for a student, then we need its name, class, etc, etc. so; it becomes a weak entity.

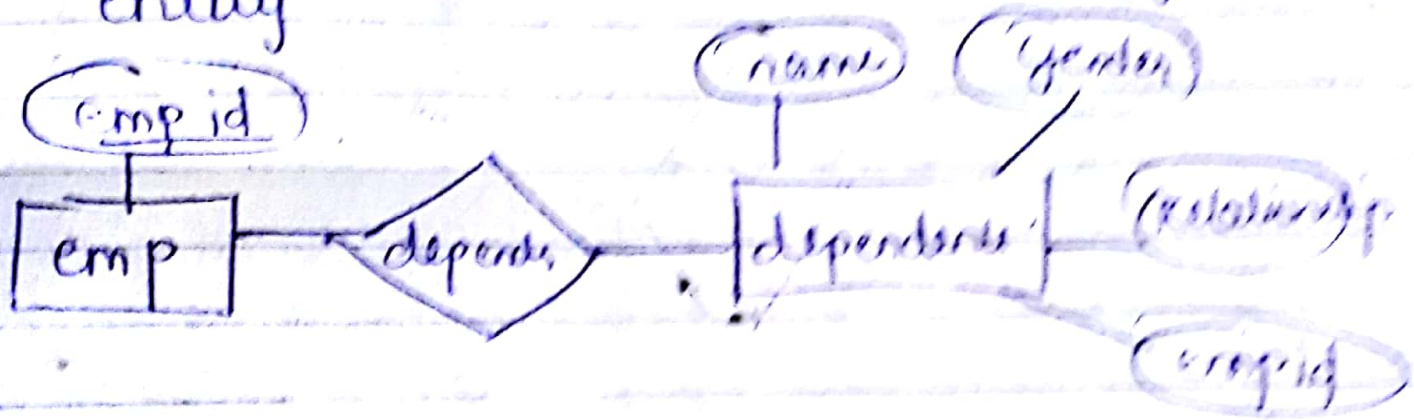
Eg;



Scanned by CamScanner

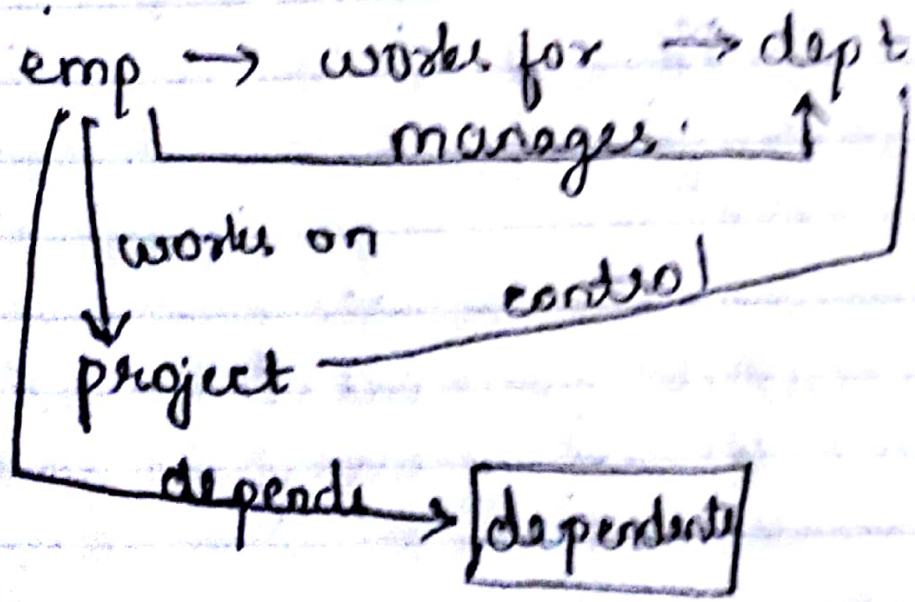
As many employee can have daughter with same name, so new dependents become a weak entity and depends weak relationship.

But if we use referential integrity then dependents become a strong entity

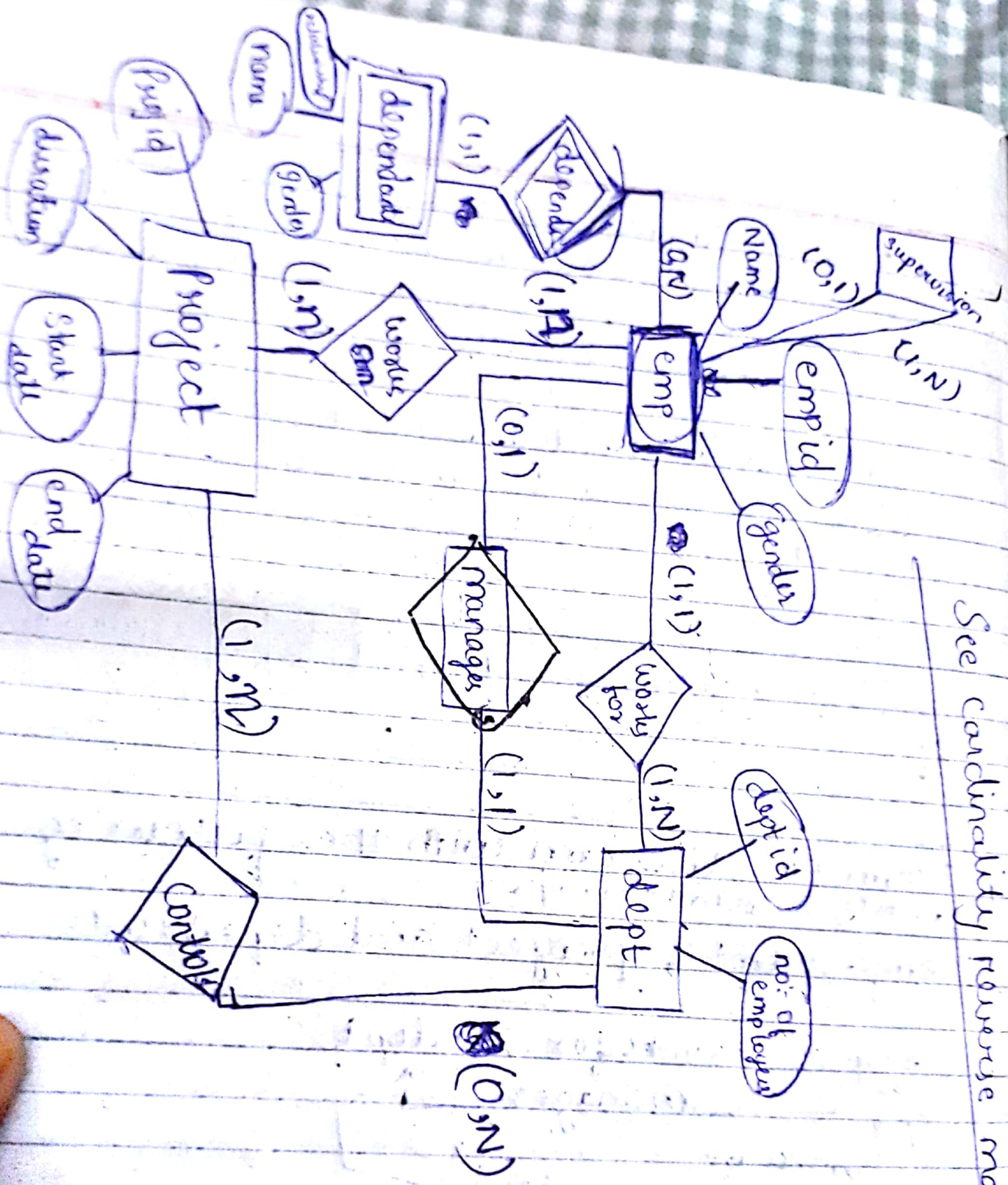


Q Draw ER diagram with the following entity & relationship:-

emp. ; dept ; project and dependents.



Scanned by CamScanner



See cardinality reverse manner

★ When we write cardinality, we write in a reverse manner.

★ (a, b)

a ↑ participation

b ↑ cardinality

Scanned by CamScanner

08/8/19

Relational Database

employee

emp-id	name	dept	Phone no	address
1	A	ABC	123	RCPUR
2	B	DEF	456	Ramchandi
3	C	GHI	789	Hanuspur

→ For every entity, we will make a table.
→ The whole is related to itself, not with other employee record
→ Table is a set of tupple
→ Complete one row is tupple or one record
→ " column is known as attributes
→ Degree of relationship (or arity) = total no:- of columns.

atmoic → that cannot be divisible
Phone no → is not atomic as it can have multiple values.
If we have a constraint that we can store only one phn no:- then it is atmoic, but if for the same case if we divide phn no:- into state code, city code then it is non-atomic.

→ Address is also non-atomic.

Scanned by CamScanner

→ Can we compare NULL & NULL ; NULL & 0 ?
We can't compare NULL & NULL as value is not known so how can we compare.

Keys :-

→ Super key → set of attributes that can uniquely identify a tupple in a table is known as super key.

* There are some set of attributes that can have same value and some set of attributes that cannot have same value.

minimal super key :- delete that key after which the remaining keys can clearly identify the tupple is minimal superkey or candidate key.

Such a name is given because they are the candidates for primary key.

* Primary key is choosen by the programmer.

Suppose emp-id & phn no. are candidate key, then we have chose emp-id as primary key as it

* Primary key value can rarely change and can never have NULL values.

Scanned by CamScanner

If we have a relation with 4 attributes then how many keys (or subsets of attributes) are there?

R(A, B, C, D)

* Default superkey → set of all attributes.

Ans $2^4 - 1 = 16 - 1 = (15)$

A, B, C, D, AB, AC, AD, BC, BD, CD, ABC, ABD, ACD, BCD, ABCD.

Suppose A is our primary key.
Then all the sets containg A are super key.
Now, Our minimal super key is A which is the primary key.

Foreign key :- In referential integerity we use our primary key that is our foreign key.

Eg.

emp (referring table)

Foreign key did	empid	name
11	1	A
12	2	B
12	3	C
12	4	D
13		

can never have NULL value.

dept (referenced table)

PK Deptid	dname
11	P
12	Q
13	R
14	S

Scanned by CamScanner

* The thing that doesn't exist, we cannot refer that thing in our referencing table

Referred table → Table whose primary key we are using in another table. That another table is known as referencing table.

* Every foreign key must have a primary key.

Contraints on FK. (Foreign Key)

PK A	B
1	A
2	B
3	C
4	D
5	D

C	D (FK)
11	1
12	1
13	2
14	3
15	5
16	4

On Delete no Action

First we will have to delete foreign key, then only we can delete primary key.

; Suppose if we have deleted PK 1; then the FK 1 (will now will act as a dangling pointer) referencing to nothing.

Scanned by CamScanner

* ~~Candidate keys can be primary key~~

Primary key can be proper subset of candidate key (not necessarily); primary key is subset of candidate key (always).

On delete cascade

→ After one deletion of the primary key what was its effect...

→ Suppose, if we delete PK 1 then in the 2nd table; the FK 1 (tupple) will be deleted.

On delete set NULL.

→ If we have delete the primary key; then foreign key will be NULL..

→ If have one constraint on FK that it cannot be NULL then first we have to delete PK and then FK.

Advantage →

→ Data won't be lost. (i.e. the whole tupple won't be deleted)...

Eg: If we have deleted PK=1; then in another table FK will have NULL value.

Q Which of the following is not a super key with VWXYZ (attributes).

V Y → Primary Key.

(i) VXYZ

(ii) VWXZ ✓ (As no Y)

iii) VWXYZ.

* Primary key can have two values (combined attributes)..

Scanned by CamScanner

Q Relation R(A, B)

A	B
1	NULL
NULL	2
3	5
1	8

(1) A is PK
(2) B " "
(3) AB " " ~~NONE~~

There is no primary key as we cannot have NULL value in any of the primary key as we cannot compare

Q Relation R(A, B, C)

AB is primary key.

(1) A can have NULL value
(2) B " " " "
(3) AB " " " "
~~(4)~~ NONE

Q

PK	FK
A	C
X 2	4
3	4
4	3
5	2 ✓
7	2 ✓
9	5 ✓
6	4

This is the same table.
and we are deleting 2.
Then which all will be deleted.

5 2 X
7 8 X
9 5 X

} Because of on delete cascade.

Scanned by CamScanner

But if these two are different table; then only 2 in 5 PK and 2 in 7 PK will be deleted and not 5 in PK as they are different table; 5 is not deleted as the primary key.

3/8/18

Q.

A	B
1	NULL
2	1
2	2
3	2

(1)

A	B
NULL	1
1	2
2	3
3	4

(2)

A	B
1	NU
2	1
3	2
4	2

(3)

Consider a relation R(A, B) ; A is the PK and B is FK referring the same relation. Then which of the above will be succesfully integrate.

3rd one is correct.

(1) is not correct as primary key.

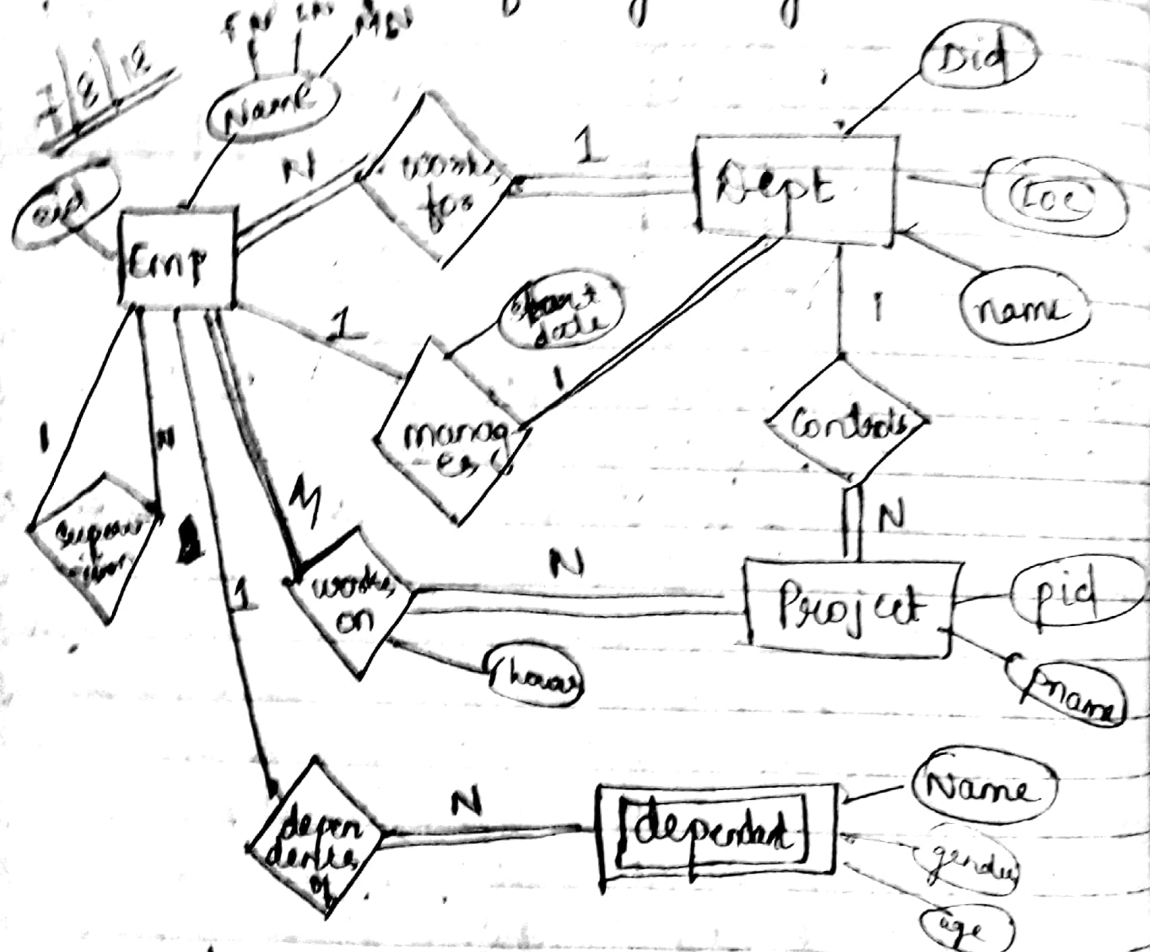
(2) is not correct as primary key can't have NULL value and FK 2 is used before 2 as P.K is inserted.

(3) 3rd is correct as FK can be NULL, and FK 1 & 2 is used after inserting PK 1 & 2.

Scanned by CamScanner

- We cannot delete primary key before deleting foreign key

→ We can insert PK (Primary key).
→ We cannot insert FK if there no PK exist.
→ We can delete foreign key



→ ER diagram shows conceptual schema.

Scanned by CamScanner

Converting ER diagram to relational schema

→ The attributes that describe relations are known as descriptive attributes. like in our case hours, start date, etc.

→ For every strong entity we will make tables first. (mapping strong entity). Primary key of entity will be the primary key

Emp (eid , fn, ln, mn, did, Seid)

Dept (did , name, Meid, startdate) [If there is a composite attribute; we will add only its further attributes). like name

Project (pid , pname, did)

→ Identify weak entity

Dependants (name, gender, age, eid)

* For 1:N relationship; entity on the 1 side of the primary key will act as a foreign key of the entity in N side.
If we will do vice-versa, then the key will have multi-value.

* Now we will see 1:1 relationship. We will see total participation!

Scanned by CamScanner

→ We will add to that entity the primary key of other entity which will act as as foreign key

→ Then we will see M:N relationships. Here we will make a table for relation, and the primary key of the both the entity will act as a primary key for this relation.

Works_On (<u>eid</u>, <u>pid</u>, hours)

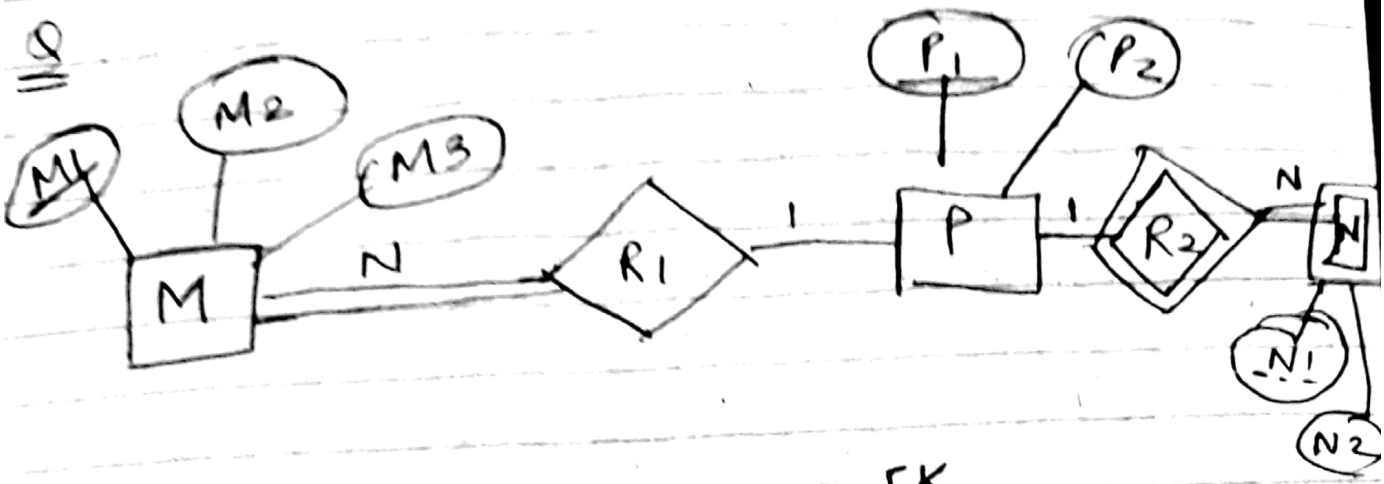
→ Then for the multivalue attribute we will make another table. Here the primary key of the entity will be added

Dept_loc (<u>did</u>, loc).

* n-ary relation → make table for every relation and every entity and add the primary key of every entity related to that relationship.
Here the number of tables will be more.
So this is effective only when we have a small database.

Scanned by CamScanner

Q



FK

M (M1 , M2 , M3 , P1)

P (P1 , P2)

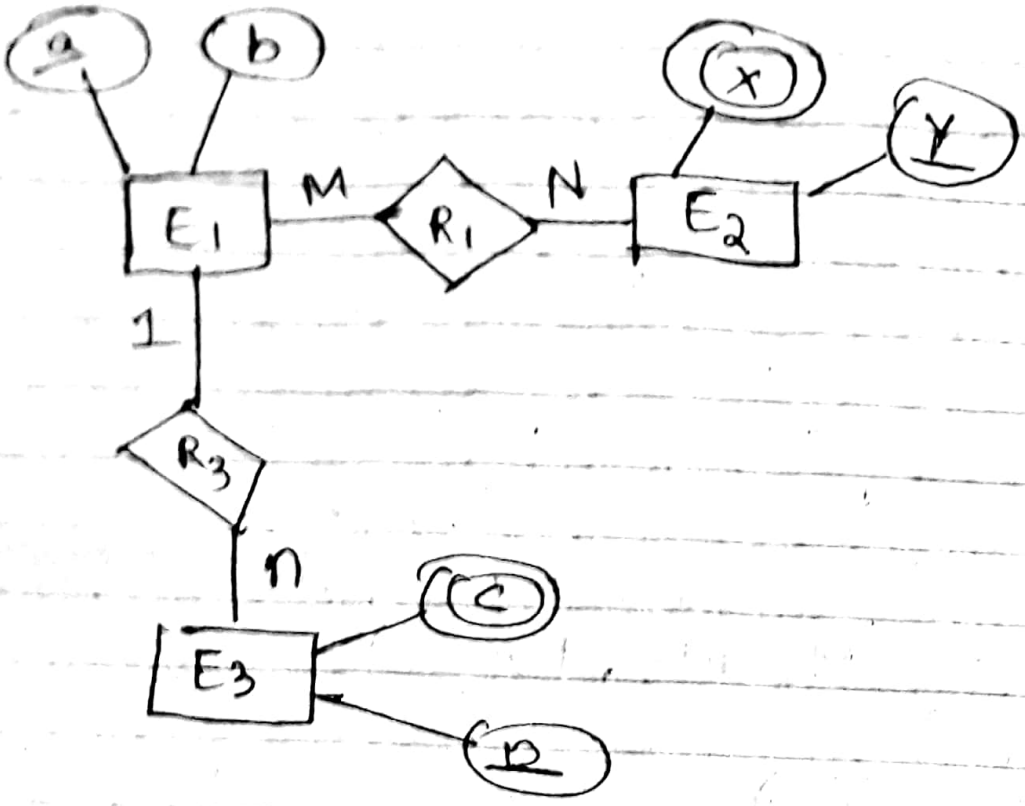
N (P1 , N1 , N2) [P1 and N1 together will act as a primary key]

Q E1 and E2 are two entities having two relationship R1 and R2
R1 is 1:n relationship
R2 is m:n "
Min no:- of tables

Ans = 3
1 for E1
1 for E2
1 for R2 (because of m:n relationship)

Scanned by CamScanner

Q



E_1 (a , b)

E_2 (Y)

E_3 (D , a)

R_4 (a , Y)

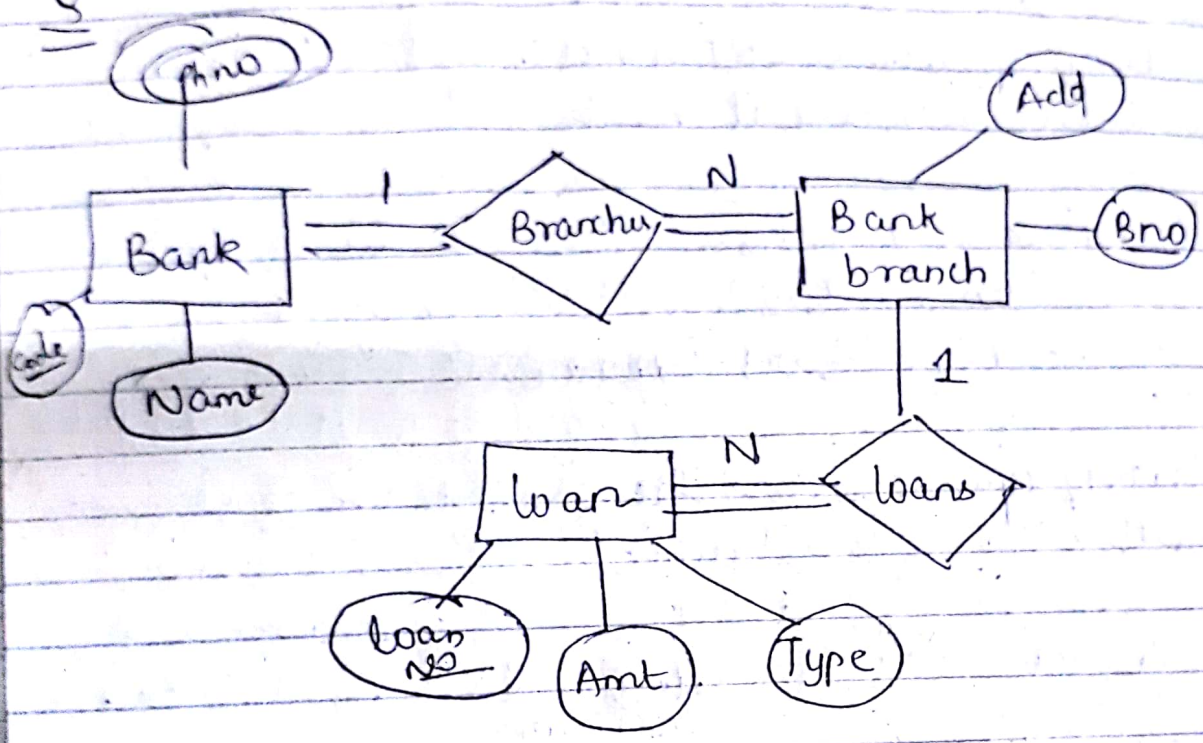
X (Y , X)

C (D , C)

Total tables = 6

Scanned by CamScanner

Q



Bank (code, Name)

Bank branch (Add, Bno, code)

loan (loan no , Amt, Type , Bno)

PhNO (code , Phno) .

Scanned by CamScanner

Relational Algebra

Way to access RDBMs or retrieve data from relational table.

Relational Algebra → is a low-level or we can say middle level language.

SQL → high level language ie, easily understandable

Unary operations → They work on one table (or single relation)

@ **Select** → $\sigma_{(\text{selection condition})}(R)$ → name of the table

→ We will apply the select on individual tupple; whenever it will match the condition; it give that tupple as output.

→ It divides our table horizontally or it partition it horizontally.

Eg:- Emp

Eid	Name	DNO	Salary
1	A	5	3000
2	B	5	4300
3	C	4	3800
4	D	1	5800

Scanned by CamScanner

$\sigma_{(dno=4)}(Emp)$

id	N	D	S
3	C	4	600

(Resulting relation)

→ Maximum tupples come in resulting relational is all.

→ Maxium attributes is all.

→ We can use all the relational operatoires here.

→ Selection is commutative

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

→ Selection is cascade.

$$\sigma_{cond1}(\sigma_{cond2}(\sigma_{cond3}(R)))$$

$$= \sigma_{cond1\ AND\ cond2\ AND\ cond3}(R)$$

(Here we can we use AND, OR, etc).

Scanned by CamScanner

Dept no is 4 and Salary is > 21000
or dept no is 5 and Salary is > 30000

Resulting relation

Empid	N	DNo	Salary
3	C	4	33000
2	B	5	43000

→ Arrity of the resulting relation = arrity of original table.
↓
(no:- of attribute).

→ Tupples of Resulting relation <= Tupples of original table.

(b) Project

π <attribute list> (R) → Table name.

Eg: π {Salary, Name} (Emp)

Scanned by CamScanner

Name	Salary
A	30000
B	43000
C	33000
D	55000

Resulting relation

→ It partition the table vertically.

→ $\pi_{name}(\sigma_{salary>3000}(R))$

Name
B
C
D

→ Resulting relation

→ $\pi_{\langle list1 \rangle}(\pi_{\langle list2 \rangle}(R))$

list1 will be a part of list2.

→ Degree of resulting relation will be less than or equal to the original relation.

→ Project may or may not be commutative.

Scanned by CamScanner

(c) Rename.

$$\Pi_{name, sal}\left(\sigma_{sal>30K}(Emp)\right)$$

Execution

Temp ← sal > 30K.

Result ← names, sal.

→ We can change attributes in the intermediate result (temp).

→ Represented as:-

① $\rho_{S(B_1, B_2, \cdots B_n)}(R)$

Here Relation name is changed from R to S.
and attribute name is changed from original to $B_1, B_2, \cdots B_n$

② $\rho_{(B_1, B_2 \cdots B_n)}(R)$ { Here only attribute name is chayd

Scanned by CamScanner

$\rho_S(R)$ { Here only relation name is changed }

Binary operations

→ Union, intersection, ~~and~~ ~~union~~ set difference and cross product.

→ It works on two tables.

→ Union compatibility → mandatory for union, intersection & set difference

Two conditions should satisfy:

→ Same degree or arrity.

→ Domain of attribute should be same.

[If name in one table is character; then in other table it should be character]

Eg: Find empid of all employees who either work in dept 5 (or) supervise an emp. who work in dept 5.

Here (or) is used so here union will be used.

If here (and) is used so here we will use intersection.

Scanned by CamScanner

R1

A	B	C	D
1			
2			
3			
4			

R2

A1	B1
11	XX
12	YY

→ Number of attributes in cross product will be m + n.

→ Number of tupples in cross product will be m * n.

A	B	C	D	A1	B1
1				11	
1				12	
2				11	
2				12	
3				11	
3				12	
4				11	
4				12	

Q List the name of dependants of each female employee.

Sol → Two tables will be used

1. emp. and and will be of dependants

2. First we will retrieve female employees from that emp table and then we will check that which female employee have dependants.

Scanned by CamScanner

Emp

Eid	Name	gend.
1	A	F
2	B	F
3	C	M

dependent

Eid	name	gender
1	X	F
2	Y	M

First Resulting relation

1	A	F
2	B	F

Now find resulting relation = (X) ← → Ans.

Eg

Books → bookid
→ title
→ publisher
→ year.

Students → stud id
→ name
→ major
→ age.

Authors → author name
→ address.

Borrows → ~~docume~~ bookid
→ studentid
→ date.

① List the year and title of each book.

Ans → $\Pi_{year, title}$ (Books).

Scanned by CamScanner

② List all info about students whose major is CS

Ans - $\sigma_{(major = CS)}(students)$

③ List all books published by MCGraw-hill before 1990.

Ans - $\sigma_{(publisher = MCGrawHill \text{ and } yearb < 1990)}(books)$

④ List the name of authors who live in Delhi.

Ans - $\Pi_{name}\left(\sigma_{(Add\ Like\ \%Delhi\%)}(authors)\right)$

⑤ List the name of students who are older than 30 & not studying in CS.

⑥

Scanned by CamScanner

• $\Pi_{name}(\sigma_{age>30})(Students) -$

$\Pi_{name}(\sigma_{major=CS})(Students)$

⑥ Rename author name in relation author to name.

$\rho_{(name, address)}(author)$

⑦ List the name of students who have borrowed books.

Two tables Students & Borrows.

Students

id	name	major	age
1	A	CS	20
2	B	ECE	30
3	C	ECE	31
4	D	CS	21
5	E	CS	32

Borrows.

Bookid	stud id	date
01	2	7/11
02	3	6/11
04	1	5/11
05	21	5/11
06	5	4/11

Scanned by CamScanner

Book id	stud id	dt	name	major	age
	2		B		
	3		C		
	1		A		
	1		A		
	5		E		

→ We will study to generate resulting relation later.

↳ Now this is doubtful (whether it is correct or wrong)

9/8/18

Q What is the need of cross product?

To find the relation between two tables; we need cross product.

→ Our resulting relation becomes too longer in cross product.

→ Join operation → are to shorten resulting relation by implementing set of conditions

① Conditional join

R ⋈ S

Eg:-

clas

ID	Name
1	A
2	B
3	C
4	D

Class info

ID	Add
1	Delhi
2	Mumbai
4	Pune

There is a neccesity that atleast one attribute should be common for join.

Cross product resulting relation.

ID	Name	ID	Add
1	A	1	Delhi
	A	2	Mumb.
	A	4	Pune.
2	B	1	Delhi
	B	2	Mumb.
	B	4	Pune.
3	C	1	
	C	2	
	C	4	
4	D	1	
	D	2	
	D	4	

⇒ We only have to retrieve when clas.id = clas info.id

↑ Condition for ⋈

Scanned by CamScanner

(after condition join)

Now our resulting relation :-

ID	N	ID	Add
1	A	1	Delhi
2	B	2	Mumbai
4	D	4	Pune

★ Here, when our condition is equal that is known as equijoin.

Eg

Emp

Eid	Ename	Dno	Salary
101	A	1	30K
102	B	2	40K
103	C	1	45K
104	D	2	90K
105	E	3	80K

Dept

Dno	Mgid
1	103
2	102
3	105

Ans T
no

Scanned by CamScanner

→ Retrieve the name of the manager of each dept.

Cross product — Emp X Dept.

Eid	Ename	Dno	Salary	Dno	Mgid
101	A	1	30K	1	103
101	A	1	〃	2	102
101	A	1	〃	3	105
102	B	2	40K	1	103
〃	〃	2	〃	2	102
〃	〃	2	〃	3	105
103	C	1	45K	1	103
〃	〃	1	45K	2	102
〃	〃	1	〃	3	105
104	D	2	90K	1	103
〃	〃	2	〃	2	102
〃	〃	2	〃	3	105
105	E	3	80K	1	103
〃	〃	3	〃	2	102
〃	〃	3	〃	3	105

$\sigma_{(Eid = Mgid)}$ (~~Cross product~~ Emp X Dept)

Ename
B
C
E

→ Ans (Resulting relation)

Scanned by CamScanner

Conditional.

→ Join is crossproduct followed by selection and then projection.

→ Equijoin is conditional join only.

Natural join.

→ Written as R * S.

→ We don't explicity mention attributes; whatever attributes have same name, it will join itself.

→ In the previous example we will compare on the basis of Dno as Dno (attribute) is same in both the tables.

→ This join will only be possible when we have some common attributes.

→ In the previous example apply natural join. (in second table there is no 3rd tupple)

Ans →

$$Emp * Dept = \left(\sigma_{(Emp.Dno = Dept.Dno)} (Emp \times Dept)\right)$$

Scanned by CamScanner

Resulting relation.

Eid	Ename	Dno	Salary	Dno	Mgrid
101	A	1	30K	1	103
102	B	2	40K	2	102
103	C	1	45K	1	103
104	D	2	90K	2	102

→ Here we will only have 5 attributes as we will not write similar attribute twice

* When there are two similar attributes, then this natural join will now become conditional join.

* In case of R * R; then all attributes are same. Now it becomes cross product.

* If we have not mentioned any operator for natural join; then by default it will take equal (=) as the operator.

Scanned by CamScanner

Eg → Teachers

Tid	Name
1	A
2	B
3	C

Courses

Cid	Cname	Tid
99	DBMS	1
100	CD	1
101	TOC	3
104	ADA	NULL

→ Here NULL values means → no teacher is assigned to ADA.

→ Query is teacher teaching subject

So resulting relation

A
A
C

→ Now here there is a subject which is not assigned a teacher and there is a teacher not teaching any subject. So there is information loss in both the table.
So, to overcome this there is Outer join.

Scanned by CamScanner

Outer join

left join

→ R ⟕ S

→ If there ~~exist~~ no a NULL value in fo
~~then it will map it~~

→ Teachers X Courses.

Tid	Name	cid	Cname	Tid
1	A	99	DBMS	1
1	A	100	CD	1
1	A	101	TOC	3
1	A	104	ADA	NULL
2	B	99	DBMS	1
2	B	100	CD	1
2	B	101	TOC	3
2	B	104	ADA	NULL
3	C	99	DBMS	1
3	C	100	CD	1
3	C	101	TOC	3
3	C	104	ADA	NULL

Scanned by CamScanner

→ Teacher ⋈ Courses.

Tid	TName	Cid	Cn	Tid
1	A	99	DBMS	1
1	A	100	CD	1
2	B	N	N	N
3	C	101	TOC	3

→ If there exist a value in left table that have no mapping, then it will map that value to the value in right table with NULL assigned to each attribute in right table.

Right outer join

→ R ⋈ S.

In ROJ we will map the NULL value in right table to the attribute in left table and anign NULL value to the left table

→ Teachers ⋈ Courses.

Tid	TName	Cid	Cn	Tid
1	A	99	DBMS	1
1	A	100	CD	1
3	C	101	TOC	3
NULL	NULL	104	ADA	NULL

Scanned by CamScanner

Full outer join

= Left outer join ∪ Right outer join

written as R ⟗ S.

Resulting relation

Tid	TName	Cid	CN	Tid
1	D	99	DBMS	1
1	A	100	CD	1
2	B	N	N	N
3	C	101	TOC	3
NULL	NULL	104	ADA	N

10/8/18

Division Operator

⇒ R ÷ S

Let say there is a table A(x,y) / B(y)

Condition is :-

① B is subset of A.

② Our output will be of x; for which every value of y will exist with that x.

Scanned by CamScanner

Eg:- A

x_1	y_1
x_1	y_2
x_2	y_1

B

y_1
y_2

Then our resulting relation

x_1 → as for x_1 there exist all values of B.

Eg # Enrolled

Sid	Cid
S_1	C_1
S_1	C_2
S_2	C_3
S_2	C_2
S_1	C_3
S_3	C_1

Courses

Cid
C_1
C_2
C_3

Enrolled ÷ Courses → S_1

* It is obvious that there will be no info of students who have not enrolled to any course

Scanned by CamScanner

Queries :

Q. Those students who have enrolled for some course. (Return Sid).

Ans. $\Pi_{Sid}(E)$

Q atleast one course :

$\Pi_{Sid}(E)$

Q Query → Who have enrolled in all course.

Ans →

$E(sid, cid) \div C(cid)$

(We have used divide operation)

Steps Involved in Division

Step ① $\Pi_{sid}(E)$ [Return all the student id that are enrolled]

O/P :

S1
S2
S3

Scanned by CamScanner

* Projection removes the duplicates.

Step2 $\Pi_{sid}(E) \times$ Courses.

S_1
S_2 X C_1 C_2 (Total tuples = 9)
S_3 C_3

Cross product shows that every student have enrolled for every subject.

Step3. Step2 – Enrolled.

(Here we will get 3 tuples)

We will get O/P as.

S_2 C_1
S_3 C_2
S_3 C_3.

This means those students who have not enrolled in all courses.

Step4 :- Π_{sid} (Step3)

O/P S_2
S_3

Scanned by CamScanner

(5) $\Pi_{sid}(E)$ – Step 4.

S_1 – S_2

S_2 – S_3.

S_3

$= S_1$ → Ans

* Whenever there all be used in the query then we will use divison

Q Student

Roll	Name
01	A
02	B
03	C
04	D

sports

Name
Badminton
Cricket
TT

Student_sports

Roll	Sport Name
02	TT
03	TT
02	Cricket
02	Bad
01	Cricket

Q1. Retrieve the name of students who play all sports

Scanned by CamScanner

Step1 $\Pi_{(Roll\ no)}$ (Student-sports)

01
02
03

Step2 $\Pi_{(Roll\ no)}$ (student-sports) × Sports

01		Bad
02	×	Cric
03		TT

Step3 Step2 − student-sports

01	Bad
01	TT
03	Bad
03	Cric

Step4 $\Pi_{(roll-no)}$ (step3)

01
03

Scanned by CamScanner

Steps

$\Pi_{(roll\text{-}name)}$ (student-sports) - Step 4

O/P = 02

Step 5.

student * step5

O/P = | 02 | B. |

Step 7

$\Pi_{(name)}$ (step 6).

O/P ⇒ B.

Till step5 we can write as.

student - sports / sports.

Scanned by CamScanner

Q. Instructor

Instid	Name	Dept	Salary

Section

Courseid	Sectionid	Semester	Year	Instid → FK

Q List the name of instructor who teaches in physics department.

Ans - $\Pi_{name}\left(\sigma_{(dept=physics)}(Instructor)\right)$

Q Find the names of instructors who work in chem dept and earn more than 90000.

Ans - $\Pi_{name}\left(\sigma_{(dept=chem\ AND\ salary>90000)}(Instructor)\right)$

Scanned by CamScanner

Q Find courseid of courses which were taught in even sem in year 2016.

$$\Pi_{(courseid)} \left(\sigma_{(semester = even \text{ AND } Year = 2016)}(section) \right)$$

Q Find the courses taught in even sem and not in odd sem.

$$\Pi_{(courseid)} \left(\sigma_{(semester = even)}(section) \right) - \textcircled{1}$$

$$\Pi_{(courseid)} \left(\sigma_{(semester = odd)}(section) \right)$$

$$= \Pi_{(courseid)} \left(\sigma_{(semester = even)}(section) - \sigma_{(semester = odd)}(section) \right)$$

Q Find the names of instruction in physics dept along with the courseid for the courses they taught.

courseid $\Pi_{(inst\_id)}$ [illegible]

Scanned by CamScanner

Instructor id in physics dept

$S_1 = \Pi_{instid}\left(\sigma_{(dept=physics)}(Instructor)\right)$

$= \Pi_{name}(S)$ (name of instructor)

$P = \Pi_{(courseid)}(Section * S_1) \longrightarrow$ Ans

(course id of the courses he took)

16/8/18

→ You have to find the possible super candidate keys for a relation R

$A_1, A_2, A_3, \ldots A_n$ attributes

① A_1 as primary key.

Suppose $n = 4$

A_1, A_2, A_3, A_4

Possible combinations

A_1 A_2 A_3 A_4

A_1A_2 A_1A_3 A_1A_4 A_2A_3 A_2A_4 A_3A_4

$A_1A_2A_3$ $A_1A_2A_4$, $A_1A_3A_4$ $A_2A_3A_4$ $A_1A_2A_3A_4$

Now our superkeys

$= A_1, A_1A_2, A_1A_3, A_1A_4, A_1A_2A_3, A_1A_2A_4, A_1A_3A_4, A_1A_2A_3A_4$

$=$ no:- of superkeys $= \boxed{2^{n-1}}$

$= 2^{4-1}$

$= 2^3 = 8$

② A_1A_2 as our primary key

Superkeys $= A_1A_2, A_1A_2A_3, A_1A_3A_4, A_1A_2A_3A_4$

no:- of keys $= \boxed{2^{n-2}} = 2^{4-2} = 2^2 = 4$

③ A_1 and A_3

~~Superkeys = A_1, A_3, A_1~~

No:- of superkeys = 12

$A_1 + A_3 - (A_1A_3)$

$= 2^{n-1} + 2^{n-1} - 2^{n-2}$

$= \boxed{2^n - 2^{n-2}}$

Scanned by CamScanner

(4) $A_1 A_3$ and $A_2 A_4$

$2^{n-2} + 2^{n-2} - 1$

$2 \cdot 2^{n-2} - 1$

$= 2^{n-1} - 1$

$= 2^{n-1} - 2^0$

Q.

Sailors (PK)

Sailorid	Sailor name	rating	age
1	A	2	18
2	B	3	19
3	C	5	20
4	D	4	20
5	E	5	21

Boats (PK)

Boatid	Boat name	Boat colour
01	AA	Blk
02	BB	Blue
03	CC	Orge
04	DD	Red

Reserves (PK)

Sailorid	Boatid	Day
1	02	Tuesday
1	03	Tuesday
4	02	Monday
5	01	Monday
5	04	Monday
3	03	Friday

(1) Find the names of the sailors who have reserved boat 03.

$\pi_{\text{Sailor name}} \left[\left(\sigma_{\text{boatid}=3} (\text{reserves}) \right) \bowtie \text{Sailors} \right]$

(2) Find the name of the sailors who have reserved Red boat.

$\pi_{\text{Sailor name}} \left(\left(\left(\sigma_{\text{boat}=\text{red}} (\text{boats}) \right) \bowtie \text{Reserves} \right) \bowtie \text{Sailores} \right)$

(3) Find the colours of the boat reserved by sailor A.

$\pi_{\text{boat colour}} \left(\left(\left(\sigma_{\text{Sailor name}=A} (\text{sailors}) \right) \bowtie \text{Reserves} \right) \bowtie \text{boats} \right)$

(4) Find the name of the sailors who have reserved atleast 1 boat.

$\pi_{\text{Sailor name}} \left(\left(\text{Reserves} \right) \bowtie \text{sailors} \right)$

Scanned by CamScanner

⑤ Find the name of the sailors who have reserved a red or a green boat.

$$\pi_{\text{sailor name}}\left(\left(\left(\sigma_{\text{boat colour}=\text{red OR bc}=\text{green}}(\text{boats})\right) \bowtie \text{Reserves}\right) \bowtie \text{Sailors}\right)$$

(or)

$$\pi_{\text{sailor name}}\left(\left[\left(\sigma_{\text{boat colour}=\text{red}}(\text{boats})\right) \cup \left(\sigma_{\text{boat colour}=\text{green}}(\text{boats})\right)\right] \bowtie \text{Reserves} \bowtie \text{Sailors}\right)$$

⑥ Find the name of sailors who have reserved all boats.

$$\pi_{\text{sailor name}}\left(\text{Reserves}_{(\text{sailor id, boat id})} / \text{boats}_{(\text{id})}\right)$$

(OR)

$$= \pi_{\text{sailor name}}\left(\pi_{(\text{sailor id, boat id})}(\text{Reserves}) / \pi_{(\text{id})}(\text{boats})\right)$$

⑦ Find the sailor id with age > 20 who have not reserved a red boat.

~~$\left(\sigma_{\text{sailor id, age}=20}(\text{sailors})\right) -$~~

~~$\left(\text{boat id, red}\right)$~~ (Reserves)

$S =$

$$\pi_{\text{sailor id}}\left(\left(\sigma_{\text{boat colour}=\text{red}}(\text{boats})\right) \bowtie \text{Reserves}\right)$$

Now with age > 20

$$\pi_{\text{sailor id}}\left[\left(\sigma_{\text{sailor age}>20}(\text{sailors})\right)\right] - S$$

Scanned by CamScanner

17/9/18

SQL

→ It is first developed by IBM and was known as sequel.

Data types:-

① Text data type

(a) char(n) :- string data type
:- fixed length

name char(20)
Anuj ---------- 20 spaces (padding)

(b) Varchar(n) :- not fixed length (variable length)

name varchar(20)
Anuj (no extra spaces) or padding

(c) BLOB → binary large object } for image, text files we use this
CLOB → complex " "

In this there is only difference in size of the file.

② Numeric data type

(a) Integer (n)

phno int(10) → size/length

(b) decimal (p, d) or numeric (p, d)

eg:- numeric (3, 1) → no. of digit to the right of decimal

44.5

③ Date-time data types

DATE YYYY-MM-DD ← (1-31)
↑ (1-12)
accept year from (1000-9999)

TIME : HH:MM:SS

DATE TIME YYYY-MM-DD HH:MI:SS

TIME STAMP (same as TIME but it define according to the system internal clock)

Scanned by CamScanner

Sql commands

① CREATE

CREATE table tablename (A_1 D_1, A_2 D_2 ... A_n D_n);

- relation :- table
- attributes - columns
- tupple - rows
- D ↑ Domain
- A ↑ attributes

* We can also add some more things like primary key, foreign key, etc.

primary key (A),
foreign key (dno) reference dept);

Eg:- CREATE table employee (eid, int(10),
ename varchar(20),
address varchar(20),
dob DATE,
primary key (eid));

We can write primary key against that attribute also like

A_1 D_1 primary key NOTNULL (that primary key can't be NULL)

Suppose, a new employee ~~dd~~ come, he don't have eid & we have to insert it in DB and we also can't give NULL value so we will give it a default value.

like

A_1 D_1 primary key default 1,

Eg2

CREATE table dept (~~dno~~ dno int(10)
dno int(10) primary key default
dname varchar(20), UNIQUE,
mgid int(10)
foreign key (mgid) References Employee(
ON DELETE ;

* We can also add check domain constraint to any attribute.

② INSERT

INSERT into tablename Values As (

* We have to add values in that sequence in which we have added attributes.

* Integer value without quotes; varchar values with double quotes.

search → if we have add 1000 tupples then how we do?

Scanned by CamScanner

④ DELETE

→ Delete from employee
• This deletes all the tupples (Only data not the schema of the table)

→ Delete from employee where eid = 5
It will delete tupple with eid = 5

→ DROP table tablename

It will delete the structure / schema of the table.

⑭ UPDATE.

→ Used to modify the stored data.
(It will change the values)

Update employee set ename = ' ',
where eid = 5;

③ ALTER.

→ This will change the schema of our table like table name, attributes, etc.

Alter table employee ADD column designation varchar(15);

Query Structure of SQL

→ Three main keywords :-
Select - from - where.
↑ ↑ ↑
attributes tablename conditions.

→ We can also write :-
select * from employee

Q Same DB with sailors, boats & reserves

① find the names and ages of all sailors

Select name, ages from sailors

* Relational algebra removes duplicates.
* SQL don't remove duplicates.
To remove duplicates we use :-

Select DISTINCT names, ages from sailors
↓
remove duplicates.

Scanned by CamScanner

② Find all sailors with rating above 7.

```
select * from sailors where rating > 7
```

③ Find the names of sailors who have reserves boat with boat id 103.

~~X = select sailor id from reserves where boat id = 103~~

~~Ans = Select name from X.~~

Ans.

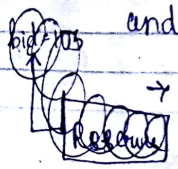
```
Select names from Sailors, Reserves
    where bid = 103 ;
```

① First it will covert the query into equivalent Relational algebra expression

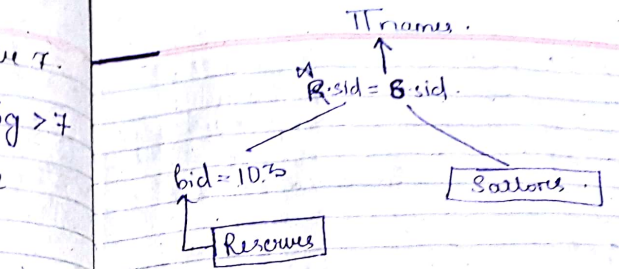
② This expression then convert to query tree and then get stored in memory and then get executed.

→ The leaf of the tree is our input table

→ First there will be condition check and then there will be join.



→ All the internal node of tree corresponds to algebraic expression.



We can write the previous que. query as:-

```
select names from Sailors S, Reserves R
    where bid = 103 AND S.sid = R.sid
```

Here we are creating alias for Sailors and Reserves as S and R; and we can use that ~~query~~ in that particular query (not in different queries).

29/8/18

Arithmetic Operations

```
select name, dept, salary * 1.1 from emp
```

→ This will create a temporary relation not permanent.

Scanned by CamScanner

→ select * from emp, dept.
It will retrieve all tupples which is cross product of emp and dept

Pattern matching

Like "% Hall %" → matches the substring

→ %B_____% → underscore is for one particular character
atleast 5 character

one _ is for one character.

→ %B_%B

_% → can be any character length name starting with B and ending with B.

→ %B_% (can be any character length name starting with B).

Between operations

→ Between 50K and 60K
(Both the end points are inclusive)

→ Not Between 50K and 60K.

In pattern (___%) → minimum 3 characters

Set operations:

① Union:

→ sql by default retains the same values
→ to remove duplicates we use keyword DISTINCT.

R	union	S	=	A
A		a_1		a_1
a_1		a_2		a_2
a_2		a_4		a_3
a_2		a_5		a_4
a_3				a_5

→ Set operations by default remove duplicates

→ If we need all duplicates in union so we will need union all

R union all S =

A
a_1
a_2
a_2
a_3
A
a_1
a_2
a_4
a_5

Scanned by CamScanner

→ For intersection → inner join or join

→ For set difference → left outer join
or except

→ Find the names of sailors who have reserved a red or a green boat.

~~Select name sailors, Reserves, boats where boat color = red union boat color = green~~

Select Sname from sailors S, Reserves R, Boats B where S.sid = R.sid and R.bid = B.bid And B.color = red or B.color = green

→ Find the names of sailors who have reserved a red and a green boat.

Select Sname . . . . . B.color red and B.color = green is wrong

→ Because this will give individual person who will reserve red boat or green boat or both

→ Second discrepancy is we are selecting according to name so there exist same name output but in AND this shouldnot exist.

60	Abc	R
70	Abc	G

So for this instead of and we can use intersection B.color red ∩ B.color green

→ Select R.sid from boats B, Reserve R, Reserve R where B.bid = R.bid

1
1
4
5
5
3

O/P.

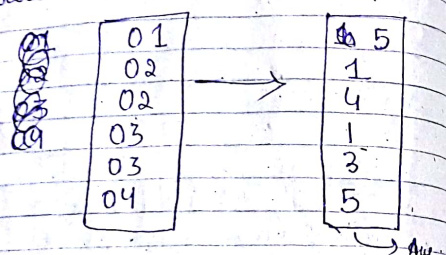
B.bid = R.bid

02
03
02
01
04
03

Scanned by CamScanner

→ Select Sid from boats ⋈ Reserves.

boats ⋈ Reserves.



→ If we are using 3 tables and doing natural join of : 1st table ⋈ 2nd ⋈ 3rd.

So this means it will first give an intermediate result of 1st ⋈ second and then with IR ⋈ 3rd.
so it is necessary that some attributes of IR should be same as 3rd table.

To avoid this we use,

Select * from (R NJ S) join B (Bid).

join according to Bid and if there is no attribute common it will give (ϕ).
So it will act as an equijoin.

→ This will be valid for both when more than one attributes are common or no attribute is common. If more than one is common, as we have written explicitly B(b,d) it will do

Ordering of tupples.

→ Order by → This is just like sorting.

Eg:- order by emp name
salary.

So by default it arrange them in ascending order.

→ If we have to use descending order then we have to explicitly mention it.

order by salary desc.

Secondary sorting.

order by emp name, salary desc.

→ In this it will arrange in descending order according to emp name (i.e, wherein comes first).

Scanned by CamScanner

29/9/17

Tutorial

Q Order(ono, amt, date, cid, sid).
Salesman(sid, name, city, commission)
Customer(cid, name, city, grade, sid).

① Prepare a list with salesman name, customer name, cities for saleman and customer who belong to same city.

Select S.sname, C.name, S.city, C.city from Salesman S, Customer C where S.city = C.city.

② Write sql statement to make a list with order number, amount, customer name and city for those orders whose amount is b/w 500 and 2000.

Select ono, amt, name, city from Order O, Customer C where amt is between 500 and 2000 AND O.cid = C.cid.

* Sql is not case sensitive.
* If in natural join there are more than one same attribute it will be same as cross prod
② Write sql query to know which salesman are working for which customer.

Select * from Salesman S, Customer C where S.sid = C.sid.

(OR)

Select * from customer a join salesman b on a.sid = b.sid.

④ Sql query for list of customers who appointed salesman with commisson more than 12%.

Select C.cid, C.name, C.city, C.grade, C.sid from customers C, salesman S where S.sid = C.sid AND commission > 12

⑤ List of customers who appointed a salesman who live in the same city where their customer lives & Commision > 12%

Select C.cid, C.name, C.city, C.grade, C.sid from customers C, salesman S where S.city = C.city AND commision > 12 AND S.sid = S.cid.

Scanned by CamScanner

⑥ Find the list of customers who either work through saleman or on their own

Select * from customers (X)

Select (all customer attributes) from Customers, Salomans where C left outer join S. (X)

⑦ Sql query to list the customers who have either placed no order or one or more order.

Select (all customer attributes) from Customers C, Orders O where O left outer join C

— X —

30/8/18

Select * from emp order by name, salary;

If name are duplicate, then to arrange the first attribute we will use the secondary attribute. This is known as secondary sorting.

Product

p-id	p-name	c-id
1	D	50
2	B	50
3	S	50
4	A	50
5	X	75

① Select * from product order by c-id desc, product name ascendig.

Ans.

5	75	X
4	50	A
2	50	B
1	50	D
3	50	S

Null value

→ It considered as unknown, generally.

A + Null = Unknown.

	AND	OR	NOT
T	U	T	U - U
F	F	U	
U	U	U	

Scanned by CamScanner

* If in where, the attribute which we are using have NULL value, then that tupple is not included in the resultant

Aggregate functions

Sum, count, max, min and average.

→ Select avg(sal) from emp where dno=5;

If there are 100 tuples and 50 of them have dno = 5, then it will calculate the avg(sal) of 50 of them.

→ Select count(*) from emp;

- It gives the number of total tupples in emp.

Aggregation with grouping

Emp

ID	Name	Dname	Sal
1	A	X	20
2	B	X	30
3	C	Y	40
4	D	Z	50
5	E	Y	60

→ Select dpt name, avg(sal) from emp group by Dname.

It will group the tupples :-

1	A	X	20	1st group
2	B	X	30	
3	C	Y	40	2nd group
5	E	Y	60	
4	D	Z	50	3rd group.

Then it will calculate the avg salary of each group.

Then it will give the avg. salary of every dept.

O/P

X	25
Y	50
Z	50

Major condition :-

All the attributes that are in select statement will also be in group by statement.

So therefore we first define attributes

Scanned by CamScanner

in group by and then define in Select. We can leave aggregate function

→ Always no of att. in select = no. in group by

Eg Select pno, pname, count(*) from Project, works on where p.pno = w.pno group by pno and pname.

Having clause :-

→ It is always used after group by

→ Only when we use group by statement then only we can use having clause

Select dpt name, avg(sal) from Emp group by Dname having count(*) >2.

O/P

X	25
Y	50

Nested queries

→ A query is embedded in another query. (That query is known as sub query or inner query)

Eg Find the names of sailors who have reserved boats 3.

Select Sname from Sailors S, Reserves R where R.sid = S.sid AND R.boatid = 3.

The same query in nested form :-

Select S.names from Sailors S where S.sid IN (Select sid from Reserves R where bid = 103).

Here the inner query will be executed first and then the outer query.

→ Find the names of sailors who have reserved a red boat.

Scanned by CamScanner

Select Snames from sailors S where S.sid IN (Select R.sid from Reserves where R.bid IN (Select B.bid from boats where B.color = red))

→ Find the names of sailors who have not reserved a red boat.

Select Snames from sailors S where S.sid IN (Select R.sid from Reserves where R.bid NOT IN (Select B.bid from boats where B.color = red))

A
C
D
E

(This end gives the name of sailors who have reserved another boat along with red)

* IN compares single value to multi value.

* If we have used NOT IN in the outer query then it will include case of the sailors who have not reserved any boat.

Do second is not correct

(Extra is only who don't reserve any boat)

A
B
C
D

→ What will be the output when we have used NOT IN in both the queries.

Select Snames from sailors S where S.sid NOT IN (Select R.sid from Reserves where R.bid NOT IN (Select B.bid from boats where B.color = red)).

B

giving doesn't reserve at all and also the one with that color boat (if he doesn't reserve any other color boat)

→ when color is taking orange it is giving → 5 C

31/8/18

Sailor

Sid	Sname	Rating
22	A	4
29	B	8
31	C	9
32	D	5
64	E	7
74	E	5
95	F	7

Boat

Bid	Bcolor
101	Blue
102	Red
103	green
104	Red

Reserves

Sid	Bid
22	101
22	102
22	103
22	104
31	102
31	103

Sid	Bid
64	101
64	102
74	103
95	102
95	104

Scanned by CamScanner

Q Find the names of the sailors who have reserved boat id = 103.

Execution of nested query

22
31
74

→ Now solve the outer query

A
C
E

→ Find the names of sailors who have reserved a red boat.

1st nested query	2nd nested query	
102	22	A
104	31	C
	64	E
	95	F

→ Final O/P.

★ Here there will be no redundancy

→ Find the names of sailors who have not reserved a red boat. (When taking NOT IN first)

1st query	2nd query	3rd query	
102	22	29 –	B
104	31	32 –	D
	64	74 –	E
	95		

→ Ans

→ It gives us the name of the sailors who have not reserved red and also who have not reserved any other boat (✓).

→ When taking NOT IN 2nd.

1st query	2nd	3rd	Final
102	101	22	A
104	103	31	C
		64	E
		74	E

(This is not correct as A has also reserved a red boat).

It is giving also the names that have reserved boat not red (even if they have reserved red). (✗).

→ Using NOT at both places.

Scanned by CamScanner

1st query	2nd	3rd	4th	Final
102	103	22	29	B
104	104	31	32	D
		64	95	F
		74		

This is giving that ids that have not reserved any boat or only red.

Co-related query

→ When the outer query and inner query are executed in co-relation

→ But in nested query, the queries are executed independently.

→ Name of sailors who have reserved boat 103.

Select S.name from sailor S where S.sid

EXISTS (Selected R.sid from Reserves R
where R.sid = S.sid and
R.bid = 103).

→ In the inner query we have written only one table but we have used both table. So it is correlated.

→ EXISTS will return true if the nested query will have some value and false when it is empty.
If it is true then it will show that in the O/P.

22 → True.
29 → False.
31 → True.
32 → False
64 → False.
74 → True.
95 → False.

22
31
74

Final O/P.

A
C
E

→ Ans.

→ If there will be Select name of sailors who have not reserved boat 103 then instead of EXISTS there will be NOT EXISTS

→ Find the names of the sailors whose rating is better than some sailor E.

Select S.name from sailor S where S.rating
> Any (Select S.rating from Sailors
where S.name = E)

Scanned by CamScanner

22 → False
29 → True
31 → True
32 → False
64 → True
95 → True
74 → False

O/P :

B
C
E
F

→ ~~Select sname~~ Find the names of the sailors whose rating is better than all sailor E.

```
Select S.name from sailor S
  where S.rating > all (Select S.
        rating from sailor S
          where S.name = E);
```

O/P :

B	C

→ Find the sailor with the highest rating.

```
Select S.name from sailor S
  where S.rating >= all (Select S.
                rating from sailor)
```

O/P = C

Scanned by CamScanner