Scanned by CamScanner

A) Consider the following schema for a Library Database:

book (book_id, title, pub_year, publisher_name)
book_author (book_id, author_name)
publisher (Pub_name, phone_no, Address)
book_copies (book_id, branch_id, no_of_copies)
book_lending (book_id, branch_id, card_no, date_out, due_date)
library_branch (branch_id, Address, branch_name)

Table creations

* Create table publicher (Pub_name varchar(20) Primary key, phone_no bigint(10), Address varchar(100));
* create table book (book_id varchar(10) primary key, title varchar(50), Not null, pub_year int, publisher_name varchar(100), Foreign key (publisher_name) references publisher (pub_name) On delete cascade);
* create table book_author (book_id varchar(10), Foriegn key (book_id) references book (book_id), author_name varchar(30));
* create table library_branch (branch_id varchar(10) primary key, address varchar(100), branch_name varchar(30));
* Create table book_lending (book_id varchar(10), branch_id varchar(10), card_no varchar(10), date_out date, due_date date, Foreign key (book_id) references book (book_id), Foreign key (branch_id) references library_branch (branch_id) On delete cascade);

Scanned by CamScanner

Consider the following schema for a Library Database:

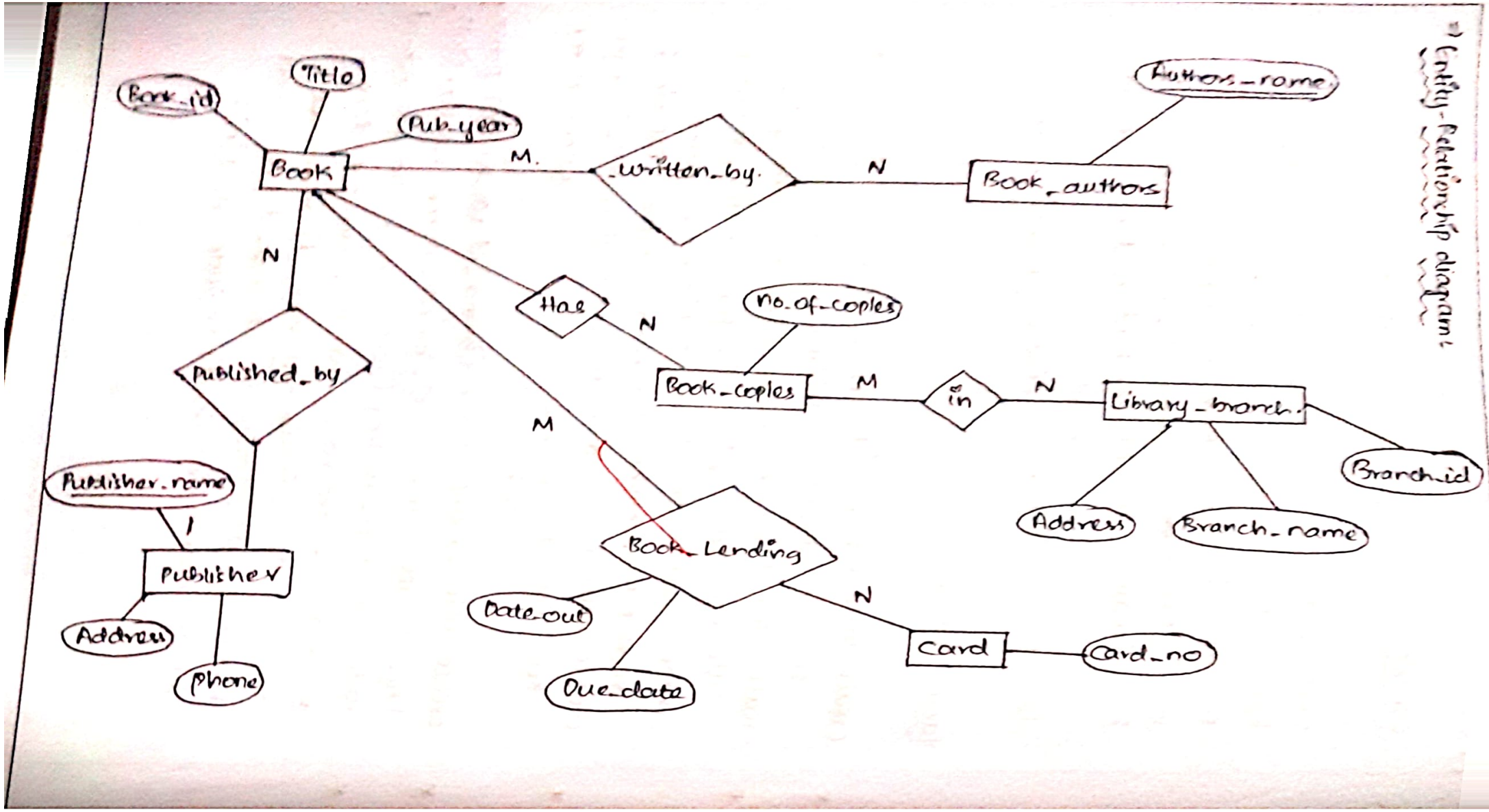
book (book_id, title, pub_year, publisher_name)
book_author (book_id, author_name)
publisher (Pub_name, phone_no, Address)
book_copies (book_id, branch_id, no_of_copies)
book_lending (book_id, branch_id, card_no, date_out, due_date)
library_branch (branch_id, Address, branch_name)

Table creation:

```
create table publisher (Pub_name varchar(20) Primary key,
    phone_no bigint(10), Address varchar(100));
create table book (book_id varchar(10) primary key,
    title varchar(50) Not null, pub_year int, publisher_
    name varchar(100), Foreign key (publisher_name)
    references publisher (pub_name) On delete cascade);
create table book_author (book_id varchar(10), Foreign
    key (book_id) references book (book_id), author_name
    varchar(30));
create table library_branch (branch_id varchar(10) primary
    key, address varchar(100), branch_name varchar(30));
create table book_lending (book_id varchar(10), branch_id
    varchar(10), card_no varchar(10), date_out date,
    due_date date, Foreign key (book_id) references
    book (book_id), Foreign key (branch_id) references
    library_branch (branch_id) On delete cascade);
```

Scanned by CamScanner

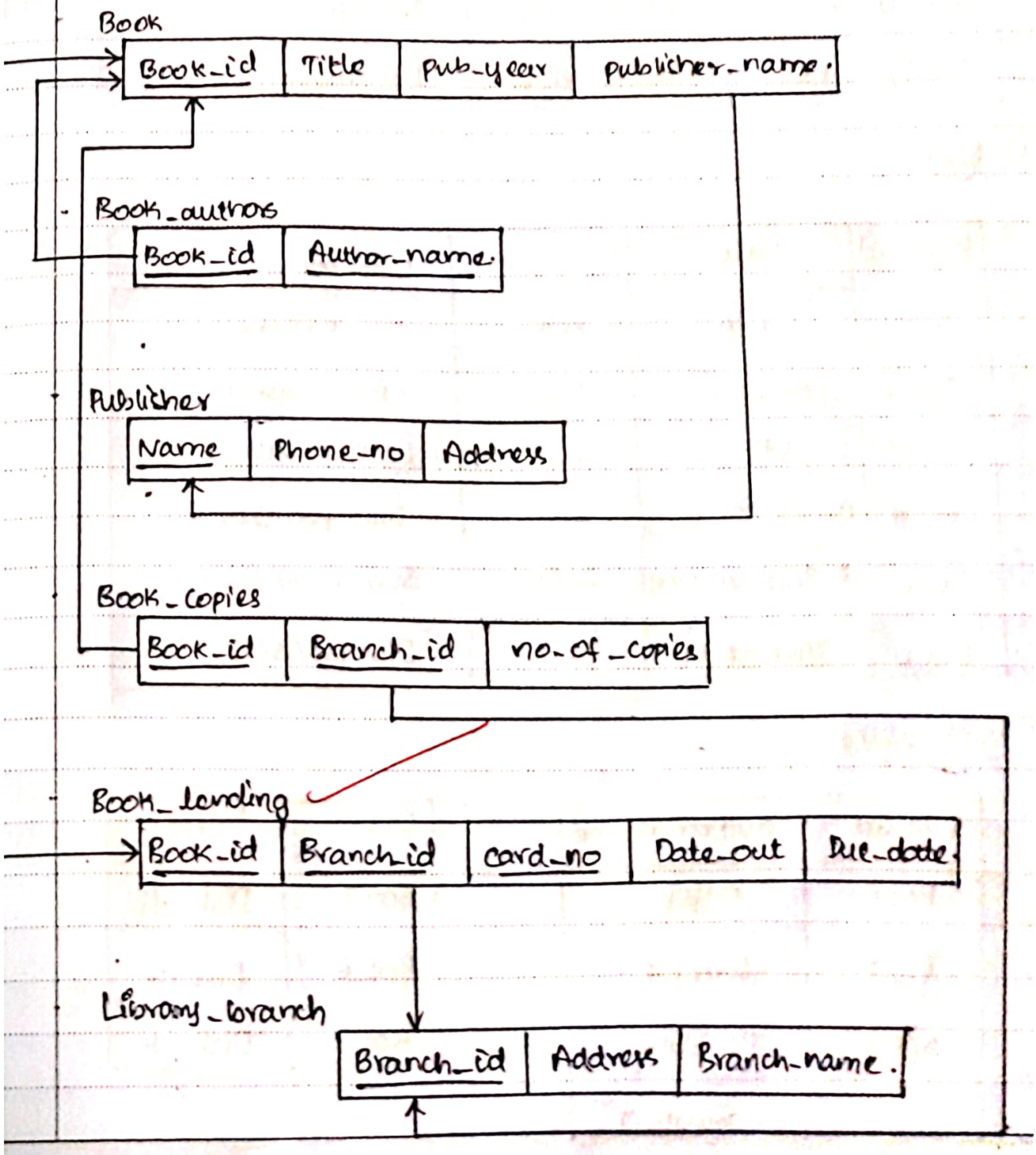
⇒ Entity-Relationship diagram:



Scanned by CamScanner

* create table book_copies (book_id varchar(10), branch_id varchar(10), no_of_copies int, Foreign key (book_id) references book (book_id), Foreign key (branch_id) references library_branch (branch_id) On delete cascade);

Schema diagram



Scanned by CamScanner

→ Publisher

Pub_name	Phone_no	Address
Dilip publisher	8312468788	Bangalore.
Suman publisher	7389225689	Bangalore
Hari publisher	6582107887	chennai
Goel publisher	7815468107	Andhra
Don Bosco	8314821870	Bangalore
CMR publisher	7821834180	Bangalore.

→ Book

book_id	title	Pub_year	publisher_name.
B001	JAVA	2018	Hari publisher
B002	PYTHON	2018	CMR publisher
B003	C#	2019	CMR publisher
B004	All IN ONE C	2019	Dilip publisher
B005	R Programming	2019	Suman publisher
B006	JAVASCRIPT	2019	Dilip publisher.

book_author

book_id	author_name
B001	Dilip.K
B002	Suman.R
B003	Haricharan
B003	Ashwath.T

B005	Brinda
B002	Disha.K
B006	Disha.K
B005	Disha.K.

Scanned by CamScanner

```
insert into publisher values ("Dilip publisher", 8312468766,
                    "Bangalore");
insert into publisher values ("Suman publisher", 7359225689,
                    "Bangalore");
insert into publisher values ("Hari publisher", 6382107857,
                    "chennai");
insert into publisher values ("Govt publisher", 7813466107,
                    "Andhra");
insert into publisher values ("Don Bosco", 8314821870,
                    "Bangalore");
insert into publisher values ("CMR publisher", 7821834160,
                    "Bangalore");

insert into book values ("B001", "JAVA", 2018, "Hari publisher");
insert into book values ("B002", "PYTHON", 2018, "CMR publisher");
insert into book values ("B003", "C#", 2019, "CMR publisher");
insert into book values ("B004", "ALL IN ONE C", 2019,
                    "Suman publisher");
insert into book values ("B005", "R Programming", 2019,
                    "Suman publisher");
insert into book values ("B006", "JAVASCRIPT", 2019,
                    "Dilip publisher");

insert into book_author values ("B001", "Dilip.K");
insert into book_author values ("B002", "Suman.R");
insert into book_author values ("B003", "Haricharan");
insert into book_author values ("B003", "Ashwath.J");
insert into book_author values ("B006", "Bindu");
insert into book_author values ("B006", "Asha.K");
```

Scanned by CamScanner

* library-branch:

branch_id	Address	branch_name.
P001	Bangalore	CMR Technical
P002	Kolkatta	Don Bosco
P003	Ahmedabad	Govt college
P004	Kerala	MVJ college
P005	J & K	ACS college
P006	Tamilnadu	CMR university.

* book-lending

Book_id	branch_id	Card_no	date_out	due_date.
B005	P003	CR002	18/05/2019	28/05/2019
B004	P001	CR009	18/05/2019	28/05/2019
B001	P003	CR001	19/01/2017	29/06/2019
B001	P004	CR003	20/11/2019	30/12/2019
B005	P001	CR003	21/11/2019	31/12/2019
B002	P002	CR002	22/12/2019	01/01/2020
B004.	P004	CR002	23/12/2019	02/01/2020

Book_copies

book_id	branch_id	no_of_copies
B005	P001	10
B001	P003	32
B003	P004	41
B001	P006	10
B006	P002	09
B002	P005	11

Scanned by CamScanner

```
insert into book_author values ("B006", "Disha.k");
insert into book_author values ("B006", "Disha.k");

insert into library_branch ("P001", "Bangalore", "CMR Technical");
insert into library_branch ("P002", "Kolkatta", "Don Bosco");
insert into library_branch ("P003", "Ahmedabad", "Govt College");
insert into library_branch values ("P004", "Kerala",
                                   "MVJ college");
insert into library_branch values ("P005", "J & K",
                                   "ACS college");
insert into library_branch values ("P006", "Tamilnadu",
                                   "CMR university");

insert into book_lending values ("B005", "P003", "CR002",
                                 "18-Aug-2019", "28-Aug-2019");
insert into book_lending values ("B004", "P001", "CR009",
                                 "18-Aug-2019", "28-Aug-2019");
insert into book_lending values ("B001", "P003", "CR001",
                                 "19-Jan-2017", "29-Jun-2019");
insert into book_lending values ("B001", "P004", "CR003",
                                 "20-Nov-2019", "30-Dec-2019");
insert into book_lending values ("B003", "P001", "CR003",
                                 "21-Nov-2019", "31-Dec-2019");
insert into book_lending values ("B002", "P002", "CR002",
                                 "22-Dec-2019", "01-Jan-2020");
insert into book_lending values ("B004", "P004", "CR002",
                                 "23-Dec-2019", "02-Jan-2020");
```

Scanned by CamScanner

book_id	title	Publisher_name	author_name	no_of_copies	branch_name
B001	JAVA	Hari publisher	Dilip.K	32	GOVT college
B001	JAVA	Hari publisher	Dilip.K	10	CMR university
B002	PYTHON	CMR publisher	Suman.R	11	ACS college
B003	C#	CMR publisher	Haricharan	41	MVJ college
B003	C#	CMR publisher	Ashwath.J	41	MVJ college
B005	R Programming	Suman publisher	Brinda	10	CMR technical
B002	PYTHON	CMR publisher	Disha.K	11	ACS college
B006	JAVASCRIPT	Dilip publisher	Disha.K	9	Don Bosco
B005	R Programming	Suman publisher	Disha.K	10	CMR technical

Scanned by CamScanner

```
insert into book_copies values("B005", "P001", 10);
insert into book_copies values("B001", "P003", 32);
insert into book_copies values("B003", "P004", 41);
insert into book_copies values("B001", "P006", 10);
insert into book_copies values("B006", "P002", 09);
insert into book_copies values("B002", "P005", 11);
```

Queries :-

Retrieve details of all books in the library i.e; book_id, title, name of publisher, authors, number of copies in each branch, etc..

```
select b.book_id, title, publisher_name, author_name,
no_of_copies, branch_name from book b, book_author ba,
library_branch lb, book_copies bc where b.book_id =
ba.book_id and b.book_id = bc.book_id and
bc.branch_id = lb.branch_id;
```

Scanned by CamScanner

→

card_no	count (*)
CR002	6.

→

book_id	title	no_of_copies.
B005	R programming	10
B001	JAVA	32
B003	C#	41
B001	JAVA	10
B006	JAVASCRIPT	9
B002	PYTHON	11

Scanned by CamScanner

Get the particulars of borrows who have borrowed more than 3 books, but from Jan 2017 to Jun 2017.

select card_no from book_lending where date_out between '09-01-2019' and '20-08-2019' group by card_no having count(*) > 3;

Create a view of all books and its number of copies that are currently available in the library.

create view bookcount as select b.book_id, title, no_of_copies from book b, book_copies bc where b.book_id = bc.book_id;

select * from bookcount;

Scanned by CamScanner

→

book_id	title	Pub_year	Publisher_name.
B001	JAVA	2018	Mari publisher
B002	PYTHON	2018	CMR publisher
B004	ALL IN ONE C	2019	Dilip publisher
B005	R Programming	2019	suman publisher
B006	JAVASCRIPT	2019	Dilip publisher.

→

book_id	title	pub_year	pub_name.
	JAVA	0	NULL

→

book_id	title	pub_year	pub_name.
B001	C#	1999	CMR publisher
B005	R Programming	1990	CMR publisher

→ Empty set.

Scanned by CamScanner

(4) Delete a book fr Book table. Update the contents of other tables to reflect this data manipulation operation

→ delete from book where book_id = 'B003';

Select * from book;

(5) create a partition of book table based on the year of publication. Demonstrate its working with a simple query.

→ create table book1 (book_id varchar(10), title varchar(20) pub_year int, pub_name varchar(30), Primary key (book_id, pub_year)) partition by RANGE (Pub_year) (Partition P1 VALUES LESS THAN(1990), Partition p2 VALUES LESS THAN (2000 partition p3 VALUES LESS THAN (2010), partition P4 VALUES LESS THAN maxvalue);

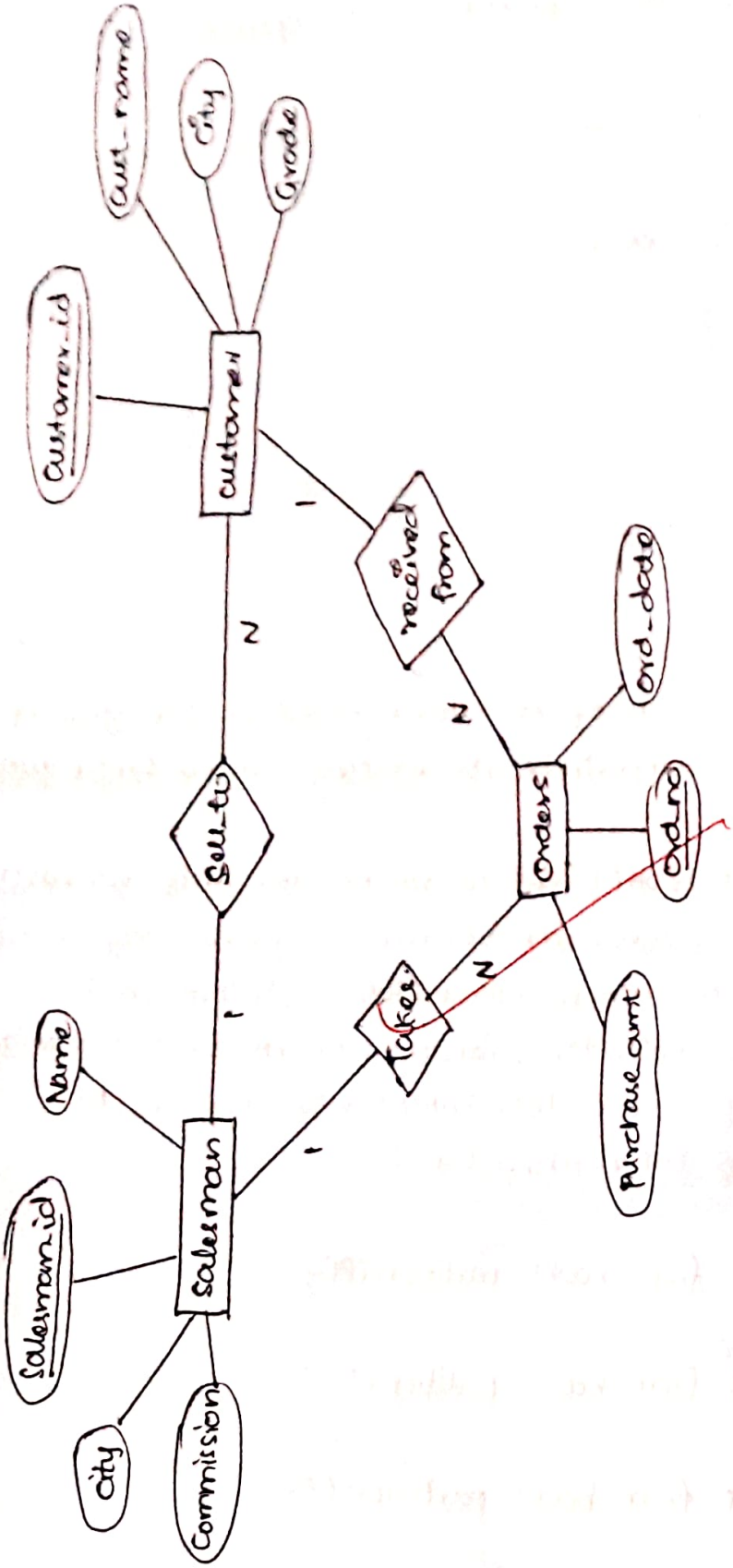
--> Select * from book1 partition (P1);

--> Select * from book1 partition (P2);

--> Select * from book1 partition (P3);

Scanned by CamScanner

→ Entity-Relationship diagram



Scanned by CamScanner

Consider the following schema for order Database:-

Salesman (salesman_id, Name, City, Commission)
customer (customer_id, cust_Name, city, Grade, Salesman_id)
Orders (Ord_no, Purchase_amt, ord_date, customer_id, Salesman_id)

Table creation:-

create table salesman (salesman_id varchar2(10), name varchar2(20), city varchar2(20), commission varchar2(20), primary key (salesman_id));
create table customer (customer_id varchar2(10), cust_name varchar2(20), city varchar2(20), grade int, primary key (customer_id), salesman_id references salesman (salesman_id on delete cascade);
create table orders (ord_no varchar2(10), purchase_amt int, ord_date date, primary key (ord_no), customer_id references customer (customer_id) on delete cascade, salesman_id references salesman (salesman_id) on delete cascade);

Scanned by CamScanner

Salesman_id	Name	city	commission
S001	Rakesh.R	Bangalore	5
S002	Haricharan	Mumbai	5
S003	Kumar.V	chennai	10
S004	Dilip.K	Bangalore	7
S005	Likith.P	Kolkatta	8
S006	Ashwath.J	Bangalore	2
S007	Disha.K	Bangalore	10

Scanned by CamScanner

Schema diagram:-

Salesman:-

salesman_id	Name	city	commission.

customer:-

customer_id	cust_name	city	grade	salesman_id

orders:-

Ord_no	Purchase_amt	ord_date	customer_id	salesman_id

Insertion:-

insert into salesman values ("S001", "Rakesh.R", "Bangalore", 5);
insert into salesman values ("S003", "Kumar.V", "Chennai", 10);
insert into salesman values ("S004", "Dilip.K", "Bangalore", 7);
insert into salesman values ("S006", "Ashwath.J", "Bangalore", 2);
insert into salesman values ("S002", "Haricharan", "Mumbai", 5);
insert into salesman values ("S005", "Likith", "Kolkatta", 8);
insert into salesman values ("S007", "Disha.K", "Bangalore", 10);

Scanned by CamScanner

customer_id	cust_name	city	grade	salesman_id
C001	Hariprasad	Bangalore	10	S007
C002	Guna	Mumbai	8	S007
C003	Brinda	chennai	9	S004
C004	Deeksha	chennai	9	S001
C005	Adithya	Bangalore	8	S007
C006	Bharath	Bangalore	9	S007
C007	Fajaz	Bangalore	5	S002
C008	Sharath	chennai	10	S007

ord_no	purchase_amt	ord_date	customer_id	salesman
OR01	7050	2019-08-19	C003	S007
OR02	700	2019-08-20	C002	S002
OR03	850	2019-08-20	C007	S004
OR04	1999	2019-08-21	C003	S007
OR05	39999	2019-09-01	C004	S004
OR06	50000	2019-09-01	C006	S002
OR07	7500	2019-09-02	C006	S007

Scanned by CamScanner

* insert into customer values ("C001", "Haripranav", "Bangalore", 10, "S001");
insert into customer values ("C003", "Rohida", "chennai", 9, "S004");
insert into customer values ("C004", "Deeksha", "chennai", 9, "S001");
insert into customer values ("C007", "Eajaz", "Bangalore", 5, "S002");
insert into customer values ("C002", "Kiran", "Mumbai", 8, "S001");
insert into customer values ("C005", "Adithya", "Bangalore", 8, "S001");
insert into customer values ("C006", "Bhavath", "Bangalore", 9, "S001");
insert into customer values ("C008", "Shravath", "chennai", 10, "S001");

* insert into orders values ("OR01", 7050, "19/08/2019", "C003", "S001");
insert into orders values ("OR02", 700, "20/08/2019", "C002", "S002");
insert into orders values ("OR03", 850, "20/08/2019", "C007", "S004");
insert into orders values ("OR04", 1999, "21/08/2019", "C003", "S001");
insert into orders values ("OR05", 39999, "01/09/2019", "C004", "S004");
insert into orders values ("OR06", 50000, "01/09/2019", "C006", "S002");
insert into orders values ("OR07", 7500, "02/09/2019", "C006", "S001");

Scanned by CamScanner

customer_count
5

Salesman_id	name
S007	Disha.K

Salesman name	customer_name	same city.
Disha.K	Haripravar	Same city
Disha.K	Adithya	same city
Disha.K	Bharath	Same city
Disha.K	Guru	Different city
Dilip.K	Brinda	Different city
Rakesh.R	Deeksha	Different city
Haricharan	Eajaz	Different city
Disha.K	Sharath	Different city

Scanned by CamScanner

Queries:

(1) Count the customers with grades above Bangalore's average.

→ select count(*) as 'customer_count' from customer where grade > (select avg(grade) from customer where city = 'Bangalore');

(2) Find the name & no of all salesman who had more than one customer.

→ select salesman_id, name from salesman where salesman_id in (select salesman_id from customer group by (salesman_id) having count(*) > 1);

(3) List all salesman's & indicate those who have and doesnt have customers in their cities using union operation.

→ (select name as "salesman name", customer_name, "same city" from salesman s, customer c where s.salesman_id = c.salesman_id and c.customer_city = s.city) union (select name as "salesman", customer_name, "Different city" from salesman s, customer c where s.salesman_id = c.salesman_id and c.customer_id != s.city);

Scanned by CamScanner

Order_no	Order_date	Salesman_id	name
0801	2019-05-19	S007	Disha.K
0803	2019-05-20	S004	Dilip.K
0804	2019-05-21	S007	Disha.K
0807	2019-09-01	S002	Haricharan
0808	2019-09-02	S007	Disha.K.

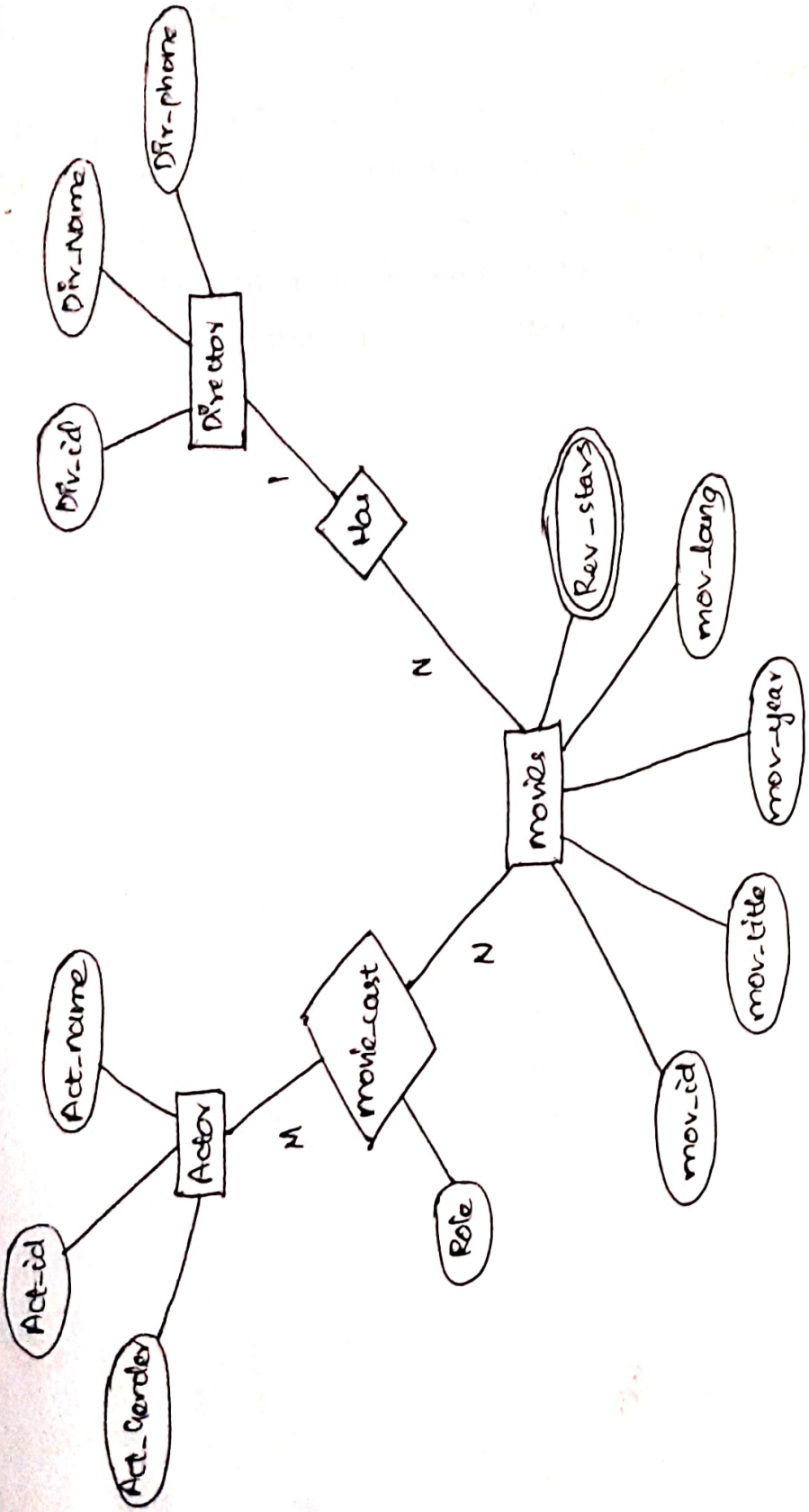
Scanned by CamScanner

4) Create a view that finds the salesman who has the customer with highest order of the day.

→ Create view highest_order as select o.order_no, order_date, s.salesman_id, s.name from salesman s, orders o where s.salesman_id = o.salesman_id and purchase_amt = (select max(Purchase_amt) from orders o1 where o.orders_date = o1.order date);

Scanned by CamScanner

Entity - Relationship Diagram



Scanned by CamScanner

Consider the schema for Movie Database:-

ACTOR (Actor-id, Act-name, act-gender)
Director (dir-id, dir-name, dir-phone)
Movies (mov-id, mov-title, mov-year, mov-lang, dir-id)
movie-cast (act-id, mov-id, role)
rating (movie-id, review, stars)

Table creation:-

create table actor (act-id varchar(10) primary key, act-name varchar(25), act-gender varchar(7));
create table director (dir-id varchar(10), primary key, dir-name varchar(25), dir-phone bigint);
create table movies (mov-id varchar(10) primary key, mov-title varchar(20), mov-year int, mov-lang varchar(15), dir-id varchar(10), foreign key (dir-id) references director(dir-id) on delete cascade);
create table movie-cast (act-id varchar(10), mov-id varchar(10), role varchar(20), foreign key(act-id) references actor(act-id), foreign key (mov-id) references movies (mov-id) on delete cascade);
create table rating (mov-id varchar(10), review varchar(20), stars int, foreign key (mov-id) references movies (mov-id) on delete cascade);

Scanned by CamScanner

Actor

act_id	act_name	act_gender
A001	Dilipk	Male
A002	Ayesha	female
A004	Deeksha	female
A005	Disha	female
A003	Haripranav	Male
A006	Haricharan	Male.

Director

dir_id	dir_name	dir_phone.
D001	Habiba	8343218700
D002	Guruprasad	9347838017
D003	Eajaz	8010124681
D004	Arun	9917778030
D005	Kumar	8317248310
D006	Likith	7829771815

movies

mov_id	mov_title	mov_year	mov_lang	dir_id
M001	Avengers Endgame	2019	English	D006
M002	Super30	2019	Hindi	D003
M003	URI	2019	Hindi	D001
M004	Gang-leader	2019	Telugu	D002
M005	Chichore	2019	Hindi	D003
M006	Fast & furious 9	2019	English	D006

Scanned by CamScanner

Schema diagram

Actor

Act_id	Act_Name	Act_Gender

Director

Dir_id	Dir_Name	Dir_phone

Movies

Mov_id	mov_title	mov_year	mov_lang	Dir_id

movie_cast

Act_id	mov_id	Role

Rating

mov_id	Rev_stars

Scanned by CamScanner

movie_cast:-

act_id	mov_id	role.
A001	M006	Villain
A002	M003	Comedian
A002	M004	Heroine
A002	M006	Comedian
A005	M002	Heroine
A003	M001	Hero
A006	M001	Villian.

Rating

Mov_id	Review	no.Stars
M003	Excellent	5
M004	Excellent	5
M001	Good	5
M005	Good	4
M003	Excellent	5
M006	Average	3.

mov_title

Avengers Endgame

fast & furious 9

Scanned by CamScanner

Queries:-

(1) List the titles of all movies directed by "Likhith"

→ Select mov_title from movies m, director d where m.dir
d.dir_id and d.dir_name = "Likhith";

Scanned by CamScanner

mov. title.

URI

Gang-leader

Fast & furious 9

act-name	mov. title	mov-year
Dilip	Fast & furious 9	2019
Ayesha	URI	2019
Ayesha	Gang-leader	2019
Disha	Fast & furious 9.	2019
Haripriyar	Avengers Endgame	2019
Haricharan.	Avengers Endgame.	2019

mov-title	max(Rev_stars)
Avengers. Endgame	5
~~Gang-leader~~	
~~URI~~	
chichore	4.
Gang-leader	5
Fast & Furious 9	3
~~Supara~~	
URI	5

Scanned by CamScanner

) Find the movie names where one or more actors acted in two or more movies.

→ Select mov_title from movies where mov_id in (select mov_id from movie_cast where act_id in (select act_id from movie_cast group by (act_id) having count(*) >1));

) List all actors who acted in movie before 2000 and also in a movie after 2015 (use JOIN operation).

→ Select act_name from actor a join movie_cast mc on a.act_id = mc.act_id join movies m on mc.mov_id = m.mov_id where m.mov_year not between 2000 and 2015;

) Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

→ Select mov_title, max(Rev_stars) from movies inner join rating using (mov_id) group by mov_title having max (rev_stars) > 0 order by mov_title.

Scanned by CamScanner

mov_id	Review	m_stars.
M003	Excellent	5
M004	Excellent	5
M001	Good	5
M005	Good	4
M003	Excellent	5
M006	Average	5

Scanned by CamScanner

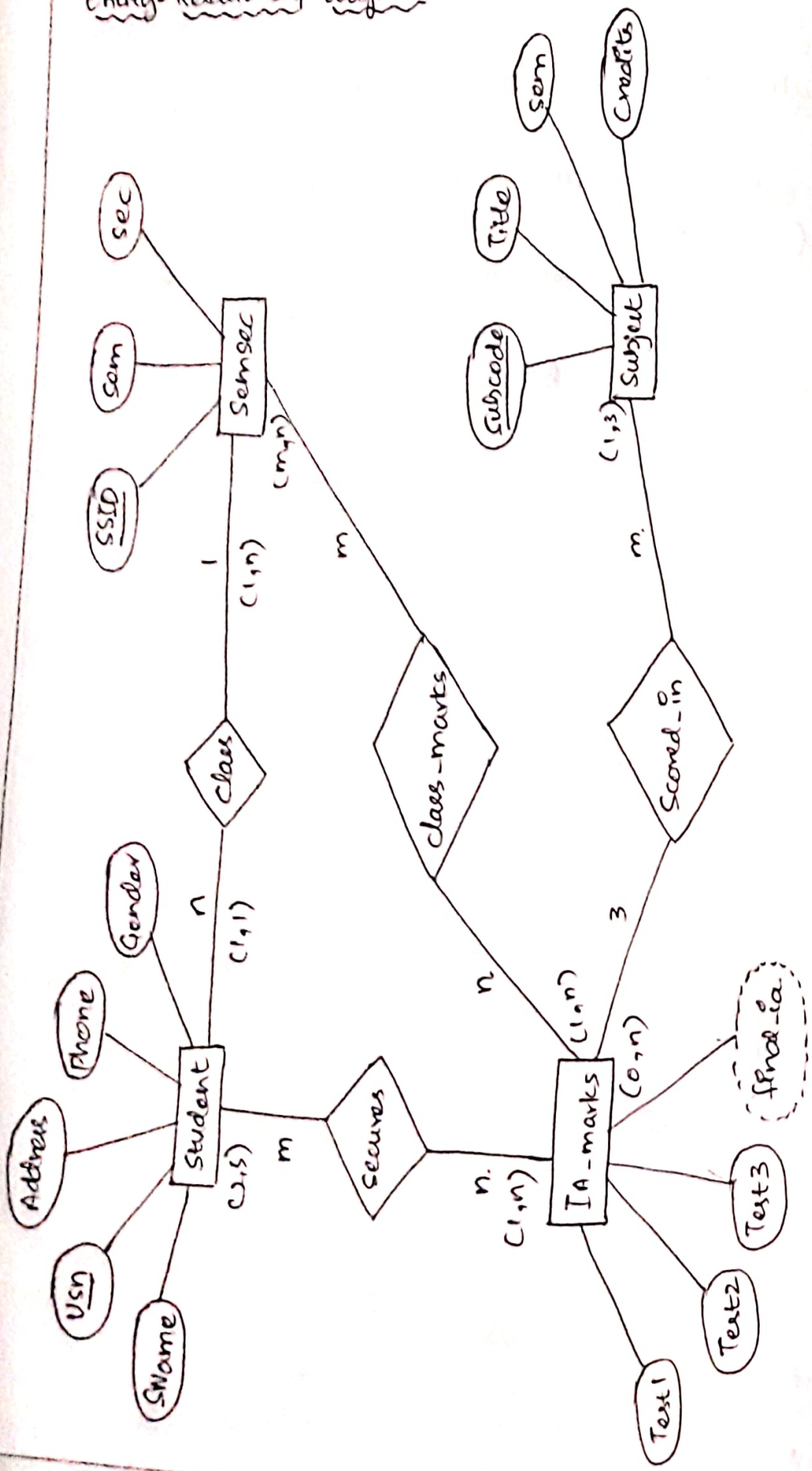
(5) update rating of all movies directed by 'Ejaz' to 5

→ update rating set rev_stars = 5 where mov_id in (select mov_id from movies where dir_id in (select dir_id from director where dir_name = " Ejaz "));

> SQL select * from Rating;

Scanned by CamScanner

Entity-Relationship diagram:-



Scanned by CamScanner

Consider the schema for college database:-

Student (usn, SName, Address, Phone, Gender)
Semsec (SSID, sem, sec)
Class (usn, SSID)
Subject (Subcode, Title, Sem, Credits)
IA_marks (usn, subcode, SSID, Test1, Test2, Test3, final_ia)

* Table creation:-

→ Create table student (usn varchar(10) primary key, Sname varchar(25), Address varchar(25), Phone number(10), Gender varchar(6));

→ create table semsec (SSID varchar(10) primary key, sem number(2), sec char(1));

→ create table class (usn varchar(10), SSID varchar(10), primary key (usn, SSID), foreign key (usn) references student (usn), foreign key (SSID) references semsec (SSID) on delete cascade on delete update);

→ create table subject (subcode varchar(8), Title varchar(20), sem number(2), credits number(2), primary key (subcode));

→ create table IA_marks (usn varchar(10), subcode varchar(8), SSID varchar(10), Test1 number(2), Test2 number(2), Test3 number(2), final_ia number(2), primary key (usn, subcode, SSID), foreign key (usn) references student (usn), foreign key (subcode) references subject (subcode), foreign key (SSID) references semsec (SSID));

Scanned by CamScanner

* Student

Usn	SName	Address	Phone	Gender
1CR17IS023	Dilip K	Bangalore	8521489512	male
1CR17IS024	Disha.K	Bangalore	7598270330	female
1CR17IS015	Ashwath	Chennai	9916663080	male
1CR17IS050	Likith	Mangalore	9606220840	male
1CR17IS034	Hari	Honnavar	7829771618	male
1CR17IS026	Eajaz	Mumbai	9164020787	male

* Semsec

SSID	Sem	Sec
3A	3	A
3B	3	B
3C	3	C
4A	4	A
4B	4	B
4C	4	C
5A	5	A
5B	5	B
5C	5	C

* Class

Usn	Class
1CR17IS024	5A
1CR17IS023	3C.
1CR17IS015	5C
1CR17IS050	3B
1CR17IS034	5A
1CR17IS026	4A.

Scanned by CamScanner

Schema diagram

Student

Usn SName Address Phone Gender.

Semsec

SSID Sem Sec

Class

Usn SSID

Subject

Subcode Title Sem Credits

IA_marks

Usn Subcode SSID Test1 Test2 Test3 final_ia.

Scanned by CamScanner

* Subject

Subcode	Title	sem	Credits
S001	Java	3	5
S002	CAD	5	7
S003	C#	4	8
S004	DBMS	3	9
S005	CN	5	8
S006	MS	3	8

* IA_marks:-

usn	Subcode	SSID	Test1	Test2	Test3	final
1CR17IS015	S001	5A	70	75	70	76
1CR17IS023	S001	3C	80	79	75	80
1CR17IS024	S003	5C	77	75	72	74
1CR17IS026	S005	3B	53	53	56	68
1CR17IS033	S006	5A	69	73	74	74
1CR17IS050	S004	4A	80	80	80	80

Scanned by CamScanner

Creating Trigger for calculating final_ia marks.

→ Create trigger finalmarks before insert on ~~ia~~ Ia_marks for each row set new.final_ia = ((new.test1 + new.test2 + new.test3) - least(new.test1, new.test2, new.test3))/2;

Scanned by CamScanner

Usn	SName	Address	Phone	Gender
1CR17IS026	Fajas	Mumbai	9164020767	male.

Sem	Sec	Gender	Count
3	A	Male/female	0
3	B	Male	1
3	C	Male	1
4	A	Male	1
4	B	Male/female	0
4	C	Male/female	0
5	A	Male	1
5	A	female	1
5	C	Male	1

Usn	Subcode	Title	Sem	Test1
1CR17IS024	S003	C#	4	77

Scanned by CamScanner

* Queries:-

(1) List all the Student details studying in fourth semester 'A' Section.

→ Select S.usn, S.SName, Address, phone, Gender from student s, semsec sm, class c, where s.usn = c.usn and sm.SSID = c.SSID and sem = 4 and sec = "A";

(2) Compute the total number of male and female students in each semester and in each section.

→ Select ss.sem, ss.sec, s.Gender, count(s.Gender) as count from student s, semsec ss, class c where s.usn = c.usn and ss.SSID = c.SSID group by ss.sem, ss.sec, s.Gender order by sem;

(3) Create a view of Test1 marks of student USN 'ICR17IS024' in all the place.

→ Create view test1-marks as select usn, s.subcode, Title, Sem, Test1, from subject s, IA-marks ia where s.subcode = ia.subcode and usn = "ICR17IS024";

Scanned by CamScanner

Scanned by CamScanner

(4) Calculate the final-ia (average of best two test marks) and update the corresponding table for all students.

→ update IA-marks set final-ia = ((test1+test2+test3) - least(test1, test2, test3))/2;

(5) Categorize the students based on the following criterian if final-ia between 80 and 100 then category is "outstanding" if the final-ia is between 50 and 80 then category is "Average" if final-ia is less than 50 category is "weak". Give the details only for the 5th semester A, B and C section students.

Scanned by CamScanner