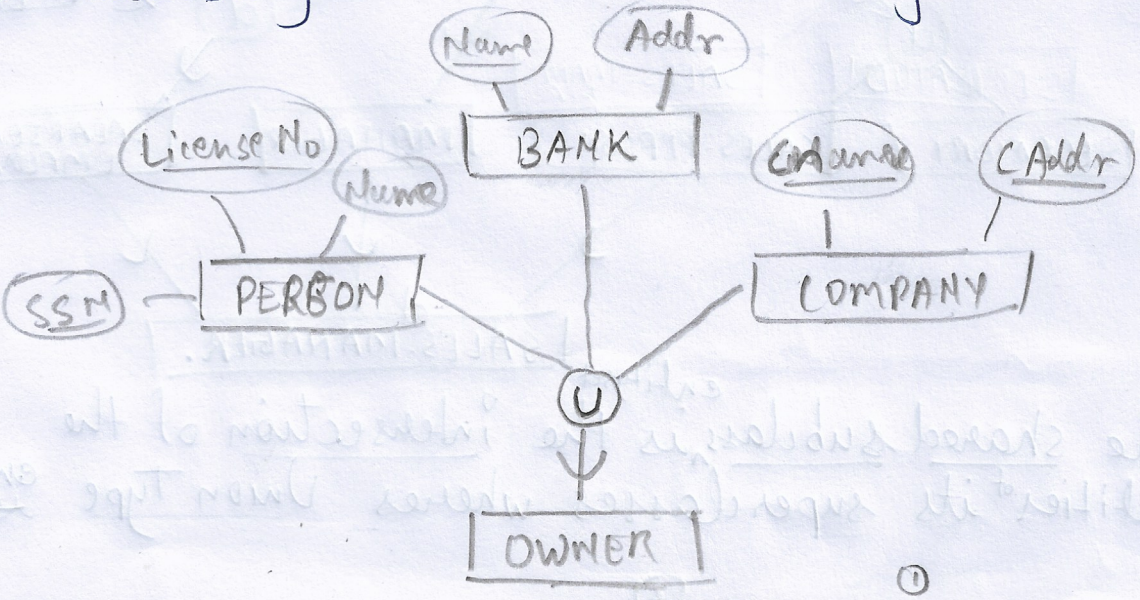
Q1. Define Union Type / Category. How is it different from shared subclass. Map the following digram to the corresponding relational database.

Ans:- A subclass that represents the collection of entities that is the subset of UNION of all distinct entities is called Union type or Category.

For example, Consider a Motor Vehicle Registration database. We have 3 entities that can play the role of vehicle owner :-

1. BANK
2. PERSON
3. COMPANY

In this scenario we can create a Union Type or Category. We create OWNER (a Union Type) that is the subclass of the UNION of the entity sets BANK, PERSON and COMPANY. All these 3 entity sets contains distinct entities which is a necessary condition for creating Union Types.

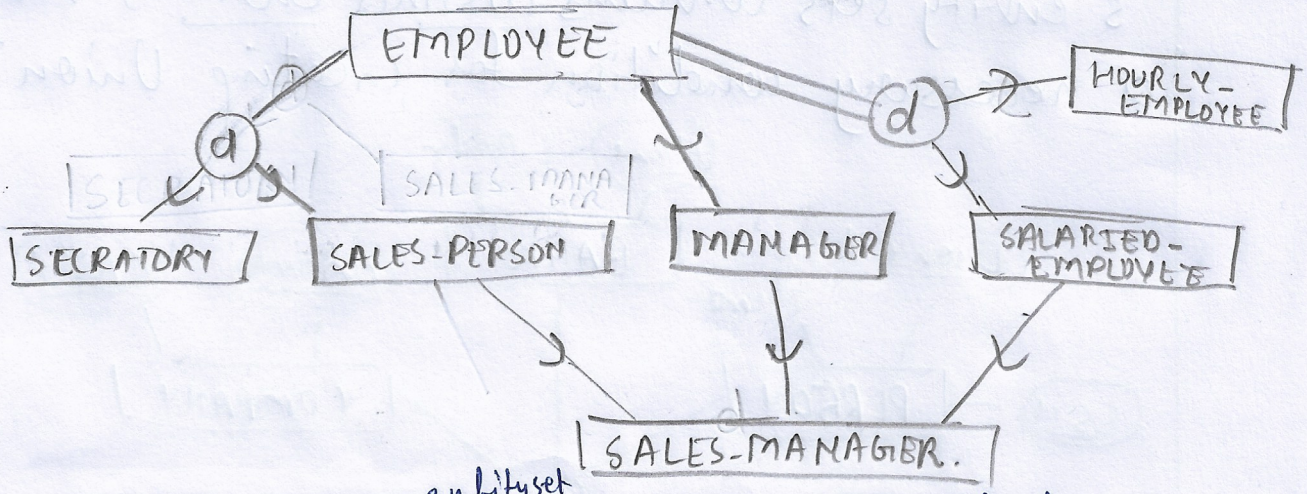


In the above EER Digram OWNER is the Union type. Any entity that belongs to the OWNER entity set will belong to any one entity set of superclasses ~~owner~~ ~~whose~~ that created OWNER.

A Union Type or Category can be total or partial. In case of total participation, the Union Type holds all the entities of all the superclasses. In case of partial participation, the Union Type holds the subset of Union of all entities in all superclasses.

⇒ A subclass with more than one superclass is called a shared subclass. A shared subclass inherits all the attributes from all the superclasses it belongs to. The concept of multiple inheritance also applies to shared subclasses.

For example, in the following EERD, SALES_MANAGER is a shared subclass.



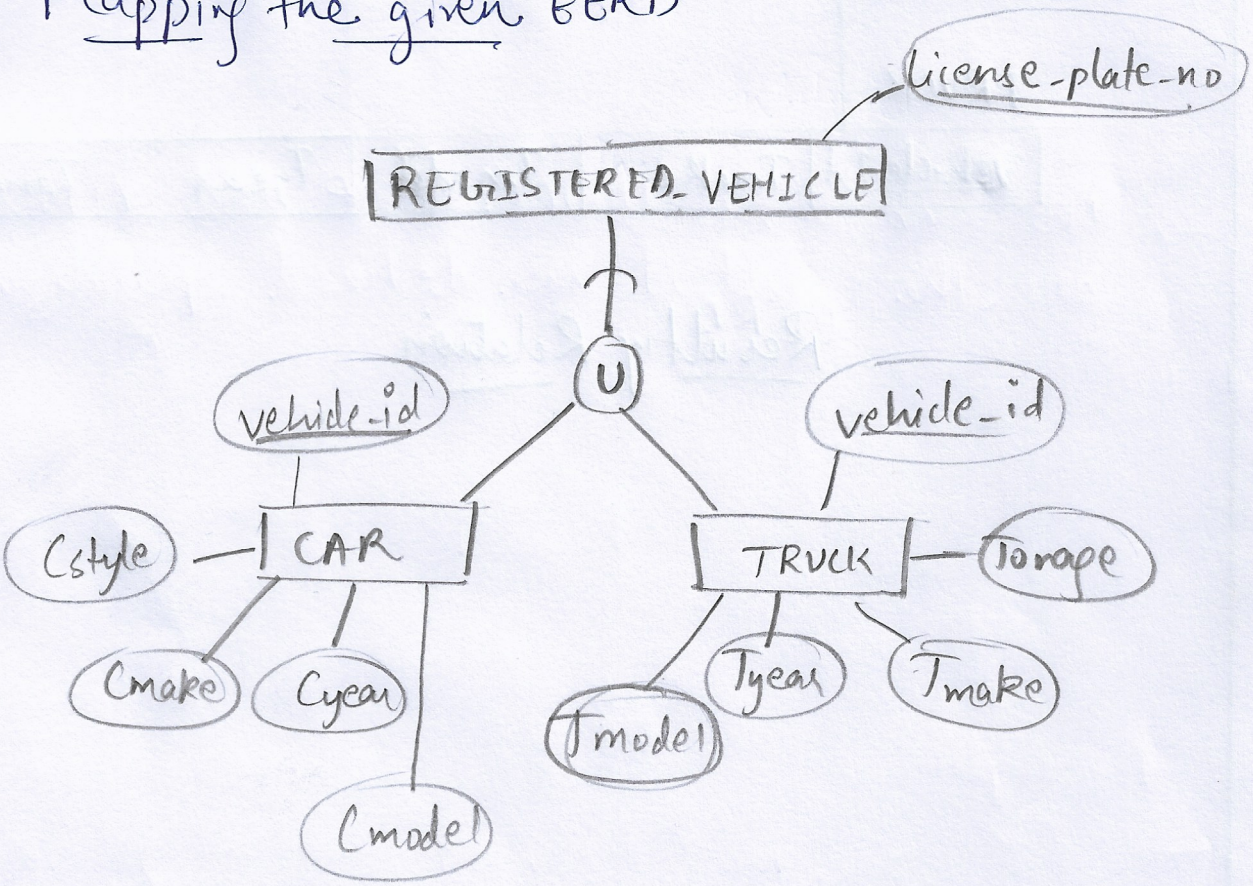
The shared subclass, entityset is the intersection of the entities of its superclasses wheres Union Type entityset ~~is the~~

is Union of entities of its superclasses.

∴ Any entity that belongs to SALES_MANAGER entity set belongs to all three entity set (superclasses) from which SALES_MANAGER is made up of.

Any entity that belongs to OWNER entity set belongs to any one of the three entity set (superclass) from which OWNER is made up of.

Mapping the given EERD



Following Step 1 to Step 8

Step-9 → Mapping of Union Type / Category

(No need of surrogate key because the superclasses of subclass REGISTERED vehicle have the same key vehicle_id)

REGISTERED VEHICLE

Vehicle_Id	license_plate_no

CAR

Vehicle_Id	Cmake	Cstyle	Cyear	Cmodel

TRUCK

Vehicle_Id	Tmake	Tmodel	Tyear	Tonage

Resulting Relation

Q2. Differentiate b/w Attribute defined and User defined Specialization. Give Example of each.

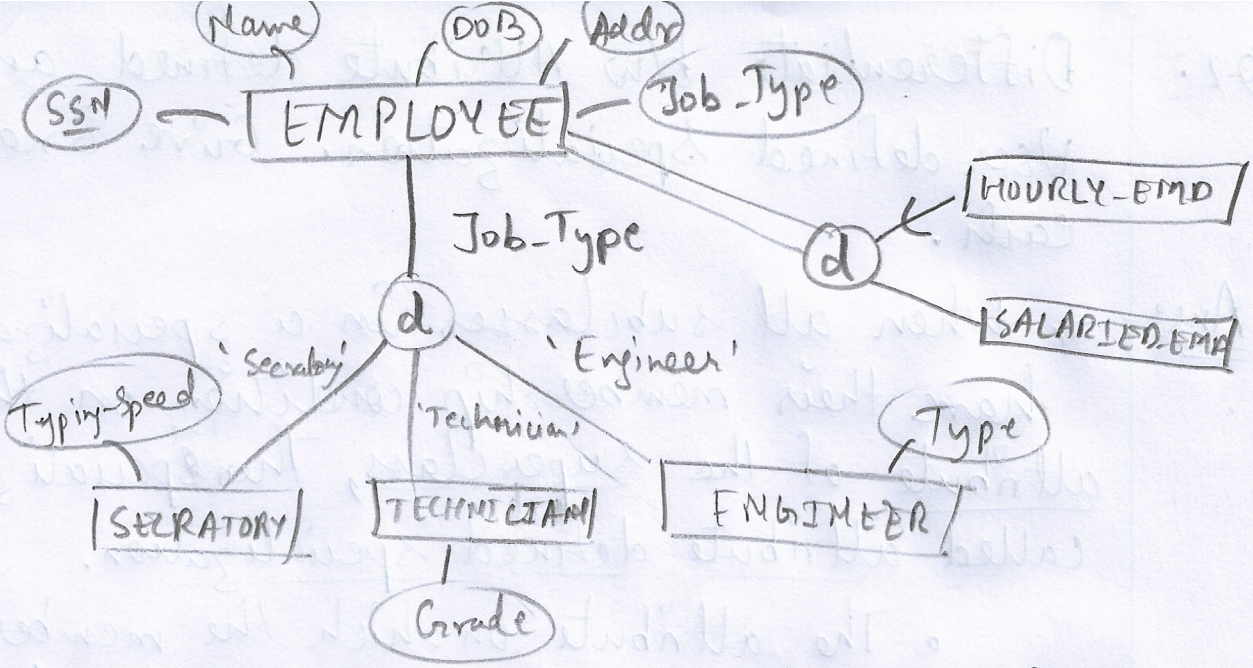
Ans:- ⇒ When all subclasses in a specialization have their membership condition on the same attribute of the superclass, then the specialization is called attribute defined specialization.

- The attribute on which the membership condition is based on is called defining attribute
- The entities with the same value of defining attribute belong to the same subclass.
- In EERD, this type of specialization is displayed by placing the defining attribute next to the arc from circle to superclass.

⇒ When all the subclasses in a specialization have their membership not based on any condition, then the specialization is called User-defined specialization

- Membership in such subclasses is based on the User, when he/she applies operation to add an entity to the subclass.
- It is specified individually ~~by each~~ for each entity by each user.

Consider the following EERDs:-



⟹ In the above EERD, we see attribute defined specialization based on the <u>defining attribute</u>

→ Job_Type

⟹ The subclasses HOURLY_EMP and SALARIED_EMP are the result of user defined specialization.

Q3. Consider a MOVIE Database in which data is recorded about the movie industry. The data requirements are summarized as follows:-

- Each movie is identified by title and year of release. Each movie has a length in minutes. Each movie has a Production company, and each is classified under one or more genres (such as horror, fiction etc). Each movie has one or more directors and one or more actors appears in it. Each movie also has a plot outline. Finally each movie has zero or more quoteable quotes, each of which is spoken by a particular actor appear in the movie.
- Actors are identified by name and date of birth and appear in one or more movies. Each actor has a role in the movie.
- Directors are also identified by name & date of birth and ~~appear~~ directs in one or more movies. It is possible for a director to act in a movie (including one he or she directs)
- Production companies are identified by name and address. A Production Company produces one or more movies.

Design an Entity Relationship digram and also specify the participation and structural constraints. State your own assumption whenever necessary.

Ans:-

After reading the requirement, we have the following entities, their attributes and relationship

* MOVIE
  - → title
  - → year_of_release
  - → length_in_minute
  - → genre (multivalued)
  - → plot_outline

* ACTOR
  - → Name
  - → DOB

* DIRECTOR
  - → Name
  - → DOB

* PRODUCTION
  - → Name
  - → Address

* QUOTES (WEAK ENTITY)
  - → actor_name (partial Key)

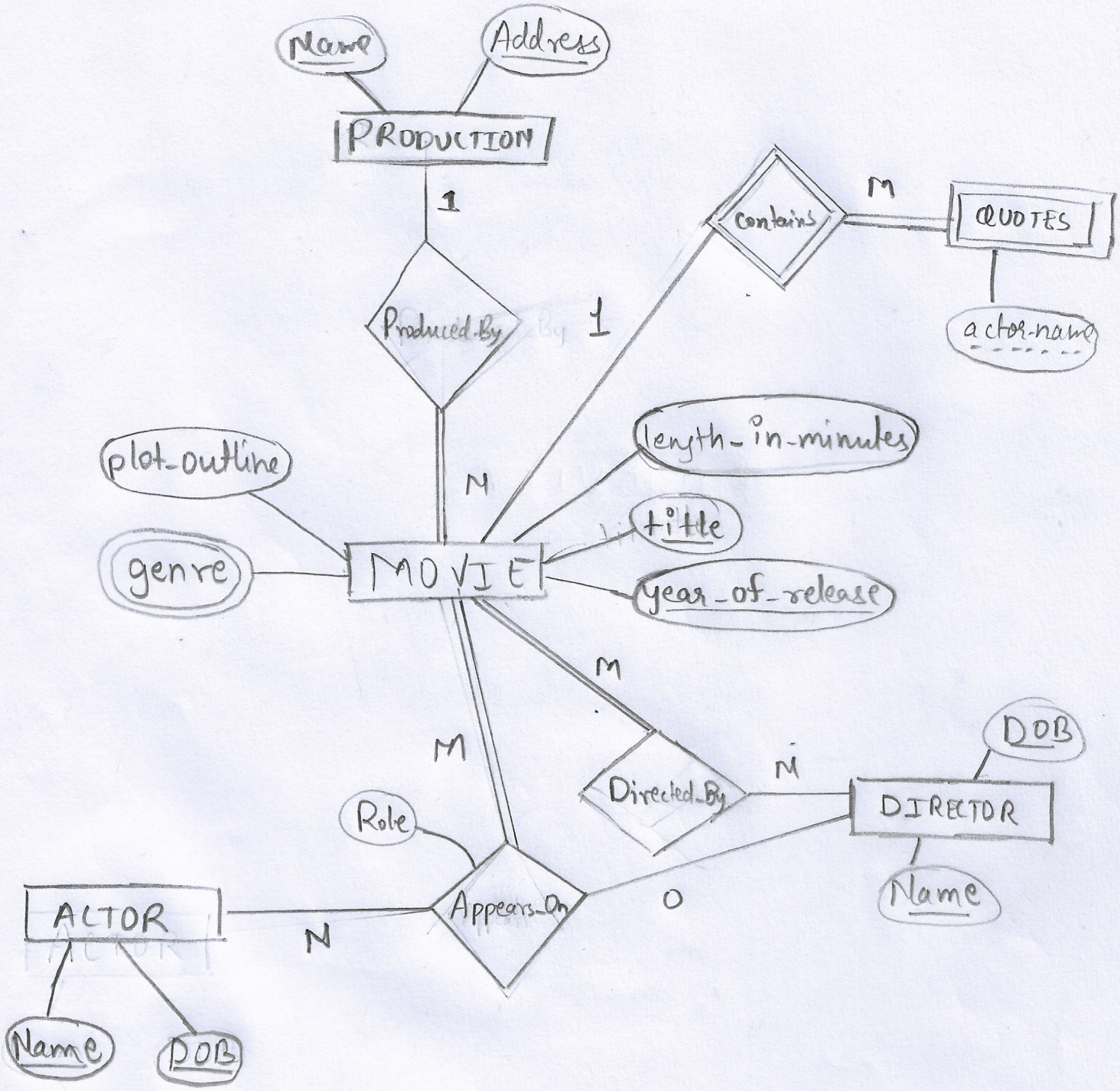
⇒ Relationships

- [MOVIE], [ACTOR], [DIRECTOR] ⇒ Appears_on
- [MOVIE], [DIRECTOR] ⇒ Directed_By
- [MOVIE], [PRODUCTION] ⇒ Produced_By
- [QUOTES], [MOVIE] ⇒ Contains.

Assumption

① Each Movie must have a Production House, a Actor and a Director

② There might be actors that don't act in movies, directors that don't direct movies and production house that don't produce movies.


Name
Address
PRODUCTION
1
Produced_By
1
Contains
M
QUOTES
actor-name
plot_outline
genre
N
MOVIE
length_in_minutes
title
year_of_release
M
M
Directed_By
N
DOB
DIRECTOR
Name
Role
ACTOR
N
Appears_On
O
Name
DOB

Q4. Consider a University database for scheduling of classrooms for final exams. This database could be modelled as consisting of classroom for final exams. This database could be modeled as consisting of following entities:

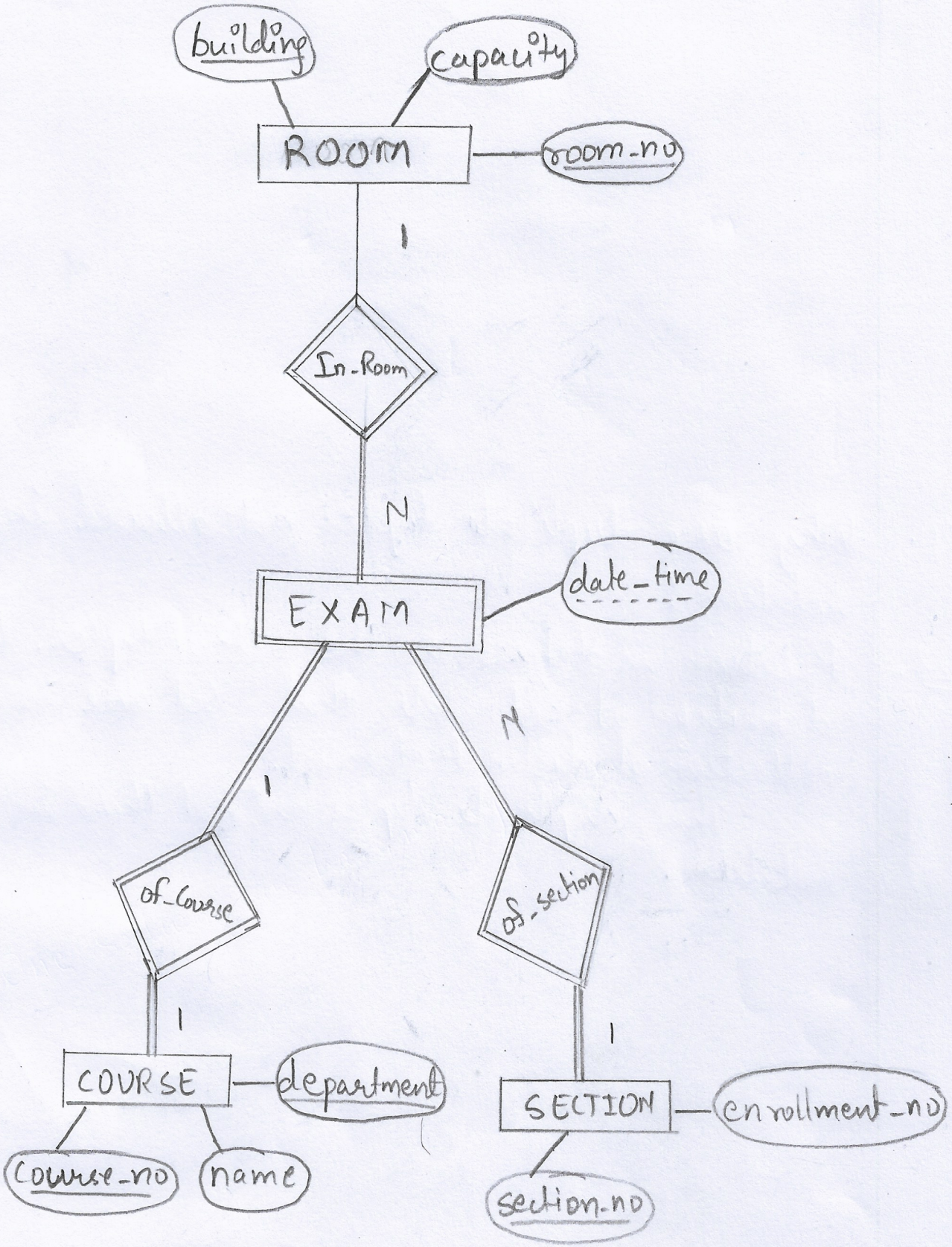
- Exam with attributes course-number, section number, room-number and date and time.
- Course with attribute course-number, name and department
- Section with attributes section-number and enrollment number
- Room with attributes room-number, capacity and building.

Draw an ER digram for the database. Also specify the participation and structural constraints. State your own assumption whenever required.

Ans:- By observing the given entities we can say that [Exam] is a weak entity with identifying entity of [Exam] being Course, Section and Room.

Assumption:

1. One Exam can happen in one room
2. There might be rooms in which no exams are held.
3. One Section can have many exams.


building
capacity
ROOM
room_no
1
In_Room
N
EXAM
date_time
N
1
of_Course
of_section
1
1
COURSE
department
SECTION
enrollment_no
course_no
name
section_no

Q5. What will be the result of the following operations on the relations M and N given below:

(i) M join M on M.X = N.P

(ii) M left outer join where M.Y = M.Q

(iii) M full outer join where M.X = M.P and M.Y = M.Q

M

X	Y	Z
5	3	6
10	7	9
5	2	7

N

P	Q	R
5	10	6
10	7	12
15	2	7

Ans:-

(i)

X	Y	Z	P	Q	R
5	3	6	5	10	6
5	2	7	5	10	6
10	7	9	10	7	12

(ii)

X	Y	Z	P	Q	R
5	3	6	5	10	6
10	7	9	10	7	12
5	2	7	5	10	6

(iii)

X	Y	Z	P	Q	R
5	3	6	null	null	null
10	7	9	10	7	12
5	2	7	null	null	null
null	null	null	5	10	6
null	null	null	15	2	7

Q6. Why are duplicate tuples not allowed in a relation?

Ans:- Relation is defined as the set of tuples. By the definition of set states that "Set ~~asse~~ is a collection of distinct element."

Thus, duplicate tuples are not there in a relation.

Q7. Consider the following database scheme and answer the following queries in SQL.

BOOK (Book-id, Title, Publisher)
BOOK_COPIES (Book-id, Branch-id, No_of_copies)
LIBRARY_BRANCH (Branch-id, BranchName, Address)
BOOK_AUTHORS (Book-id, AuthorName)
BOOK_LOANS (Book-id, Branch-id, Cardnumber, DateIssue, Due)
BORROWER (Card_No, Name, Phone, Address)

(i) Retriere the name of all the borrowers who do not have any books checkedout.

Ans:-
```
SELECT NAME
FROM   BORROWER
WHERE  Card_no NOT IN ( SELECT Cardnumber
                        FROM BOOK_LOANS)
```

(ii) Retriere the name, address and number of books issued to all borrower who have more than 5 books issued.

Ans:-
```
WITH  BOOKS_BORROWED (No, Books) AS
(     SELECT Cardnumber, COUNT(*)
      FROM   BOOK_LOANS
      GROUP BY Cardnumber )
SELECT Name, Phone, Address
FROM (BORROWER JOIN BOOKS_BORROWED ON
                Card_no = C_No)
WHERE Books > 5
```

(iii) Retriere the book titles published by 'SSS'S' but not authored by 'AAAA'

Ans:-

```
SELECT TITLE
SELECT Title
FROM BOOK NATURAL JOIN BOOK-AUTHORS
WHERE PublisherName = 'SSSS' AND
      AuthorName <> 'AAAA'
```

Q8. Consider the following relations (Key of each relation is highlighted) and answer the following in SQL.

Sales_Person (S_No, S_Name, Commission)
Product (P_Id, Description)
Sale (Date, C_No, S_No, P_Id, Qty)
Customer (C_No, C_Name, C_Address)

(i) Get the name of Sales Persons who sold the product with P_Id = 71

Ans:-
```
SELECT DISTINCT (S_Name)
FROM Sales_Person JOIN Sale ON S_No = S_No
WHERE P_Id = 71
```

(ii) Get the names of Customer who bought "Table Fans"

Ans:-
```
SELECT DISTINCT (C_Name)
FROM (Customer NATURAL JOIN Sale) NATURAL JOIN Product
WHERE Description = 'Table Fans'
```

(iii) Get the total number of products sold on "15-03-2012"

Ans:-
```
SELECT COUNT(*)
FROM Sale
WHERE Date = "15-03-2012"
```

(iv) Get the total number of products purchased by each customers.

Ans:-
```
SELECT C_Name, COUNT(*)
FROM (Customer NATURAL JOIN Sale)
GROUP BY C_No
```

Q9. Consider the following WORKER relational schema.

Worker (Worker_ID, fname, lname, salary, joining_date, department)

Bonus (Worker_ID, Bonus_Date, Bonus_amt)

Title (Worker_Id, Title, Affected from)

Ans:-

(i) Retrieve the name of employee having the highest salary in each department.

Ans:-

```
WITH SALARY_DEPT (Dept, Max_Salary) AS
(   SELECT Department, Max(Salary)
    FROM Worker
    GROUP BY Department)
SELECT fname, lname
FROM Worker, SALARY_DEPT
WHERE Dept = Department AND Salary = Max_Salary
```

(ii) Find the list of employee with same salary.

Ans:-

```
WITH SALARY_COUNT (Salary_1, Number) AS
(   SELECT Salary, COUNT(*)
    FROM Worker
    GROUP BY Salary)
SELECT (Worker_Id, fname, lname, Salary, Joining_date,
        department)
FROM Worker, SALARY_COUNT
WHERE Salary = Salary_1 AND number <> 1
```

(iii) Retrieve the departments that have less than five people in it

Ans:-
```
WITH DEPARTMENT_PEOPLE (Dept, People) AS
(   SELECT department, COUNT(*)
    FROM Worker
    GROUP BY department)
SELECT Dept
FROM DEPARTMENT_PEOPLE
WHERE PEOPLE < 5
```

Q10. Give Three examples where use of null value would be appropriate.

Ans:- NULL values are used in case where a problem occurs in adding a value to attribute. There are 3 cases when using null values would be appropriate:

(1) not applicable :- A particular entity doesn't have value for some of its attributes.

For Example BOOK entity might have an attribute e-book-price, but there exists BOOK entities that don't have a ebook version. Thus, NULL can be used in this case.

(2) missing :- A attribute value exits, but it is missing or is not provided.

For Example, consider a Music database with SONG entitity that has an attribute release date, but there exists some SONG entities that were released ~~before~~ during 1500s etc that don't have a release date. Thus, NULL can be used in this case.

(iii) not_known :- This case occur when it is not known if the attribute value exists.

For Example, a PERSON entity have a home_phone attribute, but the person doesn't have a home phone. Thus, NULL can be used here.