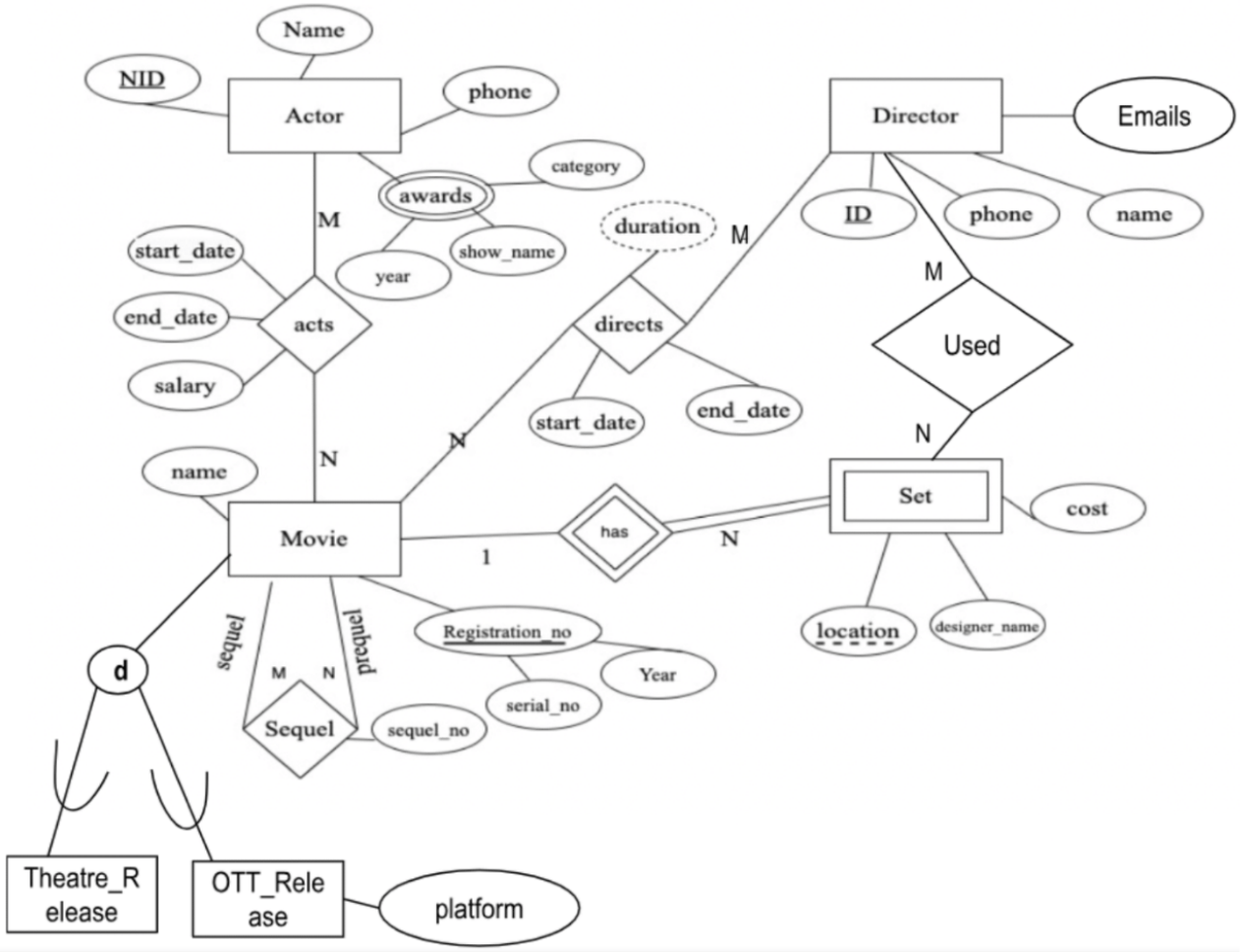
Brac University
Department of Computer Science and Engineering

Examination: Final
Duration: 1 Hour 40 Minutes
(+10 Mins for Submission)

Semester: Fall 2023
Full Marks: 45

CSE 370: Database Systems

Answer ALL of the following questions. Understanding the question is part of the exam.

1. [CO3] Construct a relational Schema by mapping the following EER diagram for a Movie Management System. For the specialization/generalization portion, choose any applicable option except 8A: separate tables for subclasses and superclasses. 	**13**

2. [CO4] Consider the following relation: **10**

AccommodationService (<u>customerID</u>, <u>flatID</u>, <u>ownerID</u>, customerName, customerContactNo, flatAddress, size, startDate, endDate, rent, ownerName, ownerContactNo, ownerNomineeID, ownerNomineeName, ownerNomineeContactNo, agreementID)

The primary key of the relation is underlined. The relation has the following additional functional dependencies (FDs):

FD1: customerID → customerName, customerContactNo
FD2: flatID → flatAddress, size, startDate, endDate, rent
FD3: ownerID → ownerName, ownerContactNo, ownerNomineeID, ownerNomineeName, ownerNomineeContactNo
FD4: size, startDate, endDate → rent
FD5: ownerNomineeID → ownerNomineeName, ownerNomineeContactNo

a. **Explain** if the above relation is in the first normal form (1NF) or not? If not, **apply** 1NF normalization. **2**
b. **Explain** if the relation(s) of no (a) is/are in the second normal form (2NF) or not? If not, **apply** 2NF normalization. **4**
c. **Explain** if the relation(s) of no (b) is/are in the third normal form (3NF) or not? If not, **apply** 3NF normalization. **4**

3. [CO5] Consider the following relational database schema for a library management system.

Book

<u>copyNo</u>	ISBN	title	edition	year	price	available

Borrower

<u>borrowerNo</u>	borrowerName	borrowerAddress	country

BookLoan

<u>copyNo</u>	<u>dateOut</u>	dateDue	borrowerNo

[**Foreign Key**: copyNo references Book (copyNo) and borrowerNo references Borrower (borrowerNo)]

The primary keys are underlined and foreign keys are mentioned in bold under each table that has any foreign keys.

Write appropriate SQL statements for the following questions (for each question write a **single query**):

a. Retrieve all the Book title, ISBN, year and price sorted by ascending order of year. If the book is published in the same year, then sort based on the price from high to low. **2**
b. Retrieve the number of books that are loaned by each borrower. For each borrower, print the borrowerNo and the number of books if the borrower loaned more than 5 books. **2**
c. List all the books published in 2016, whose price is more than all the books published in 2014 and 2015. **3**
d. Retrieve the average price of all books that have been borrowed by a borrower whose address contains the phrase ‘Baker Street’. **3**

4. [CO6]

a. **Construct** a B+ tree of order n = 4 for the following search key values inserted in the given order: 80, 70, 90, 95, 75, 85, 98, 72, 78, 88, 93, 100**.** Each time there is a split, **a new B+ tree must** be drawn. **7**

b. **Construct** a hash index on attribute "Product_Code" of the "Product" table. The hash index has 5 buckets, each capable of holding a maximum of 2 index entries. Bucket overflow is resolved using forward chaining. **5**

The "Product" table is provided below:

Product_Code	Name	Price
YN5	Laptop	999.99
QA6	Smartphone	499.99
FJ7	Smart TV	799.99
JZ6	Wireless Headphones	129.99
WW4	Digital Camera	349.99
NF7	Coffee Maker	79.99
SM8	Bluetooth Speaker	59.99
VA9	Fitness Tracker	149.99

ASCII Value Chart

Character	Value	Character	Value	Character	Value	Character	Value
0	48	9	57	I	73	R	82
1	49	A	65	J	74	S	83
2	50	B	66	K	75	T	84
3	51	C	67	L	76	U	85
4	52	D	68	M	77	V	86
5	53	E	69	N	78	W	87
6	54	F	70	O	79	X	88
7	55	G	71	P	80	Y	89
8	56	H	72	Q	81	Z	90

Steps involving hash function is summarized below:

- Find the sum of the ASCII values for each character in the given Product Code string.
- Square the sum. Extract the two middle digits from the squared result. If the squared result has n digits then you take the (n // 2)-th and ((n // 2) +1)-th digits.
- Then, calculate the sum of the two extracted digits and take the remainder after dividing the sum with the number of buckets in the hash index.

Consider the example below:

- For Product_Code = **'GX2'**, the ASCII values of the corresponding characters, 'G' = 71, 'X' = 88 and '2' = 50. The sum of the individual ASCII values = 71 + 88 + 50 = 209.
- Square of the sum 209 = (209 * 209) = 43681. This result contains n = 5 digits. So the (5// 2)-th and ((5 // 2) +1)-th digits of the squared sum are 3 and 6 respectively. Sum of the middle digits = 3+ 6 = 9. Remainder = 9 % 5 = 4. So the index entry of **'GX2'** will be stored in bucket 4.