CITY UNIVERSITY OF HONG KONG

Course code & title	:	CS3402 Database Systems
Session	:	Semester B 2023/24
Time allowed	:	2 Hours

This paper has 12 pages (including this cover page).

1. This paper consists of **FIVE** questions.
2. Write down your answer in the space provided.

*This is an **open-book** examination.*

Candidates are allowed to use the following materials/aids:

Printed lecture notes, personal notes, textbook and other course handout materials.

Materials/aids other than those stated above are not permitted.

No Electronic devices.

STUDENT ID		**VENUE**	
NAME		**SEAT NO**	

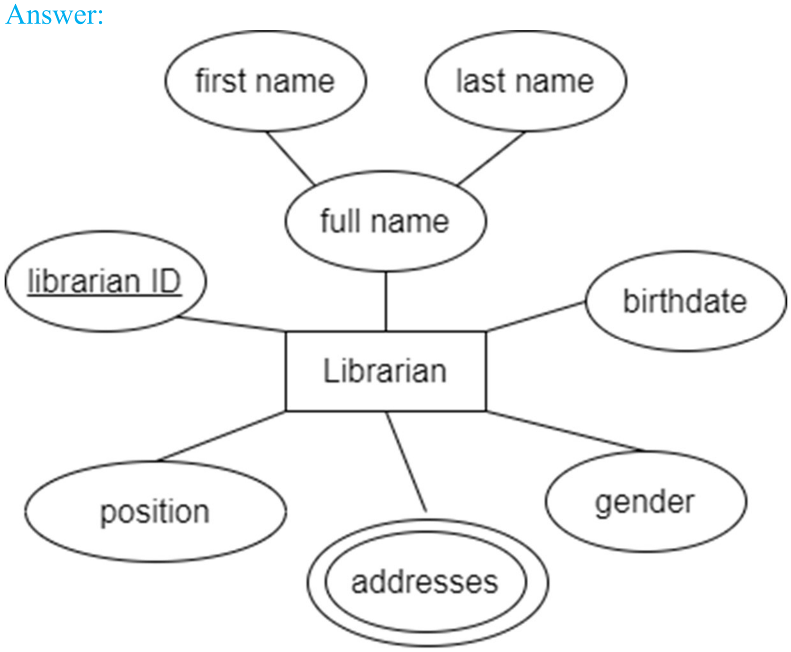
Q1 (20%)	Q2 (20%)	Q3 (20%)	Q4 (20%)	Q5 (20%)	Total (100%)

Problem ONE: ER Diagram [20 points]
Consider a library management system database that consists of the following entities: (a) **Librarian**, which has a unique librarian ID and other attributes such as full name (composed of a first name and a last name), birthdate, gender, position, and multiple addresses. (b) **Reader**, which has a unique reader ID, a unique username, and other attributes like gender, membership type, membership duration, and the number of books currently borrowed in total. (c) **Book**, which has a unique book ID and other attributes like book title, authors, publisher, genre, and the number of total copies. (d) **Book Copy**, which has attributes including the ID of the copy, the current status (available/borrowed), and the arrival date of the copy in the library.

The database also keeps track of three relationships: (a) **Has**, which describes which book has which book copies. (b) **Manage**, which describes which librarian manages which book during a specific period. (c) **Borrow**, which describes which book copy is currently borrowed by which reader, as well as the borrowing date and returning date of the book copy for the reader. Based on the above description, please answer the following questions about the ER diagram of this database:

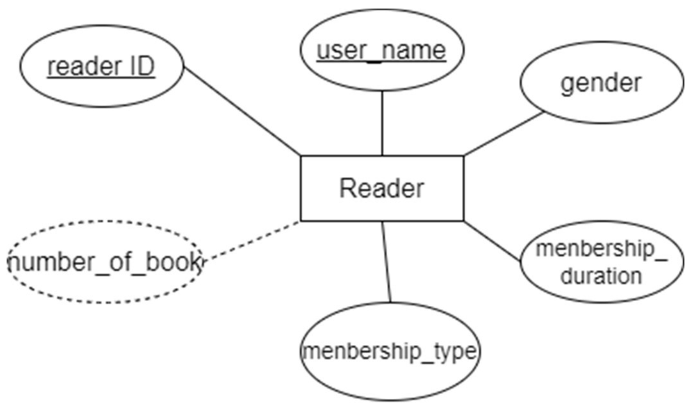
1. Please draw the ER diagram for the entity type **Librarian**. [4 points]

   Answer:

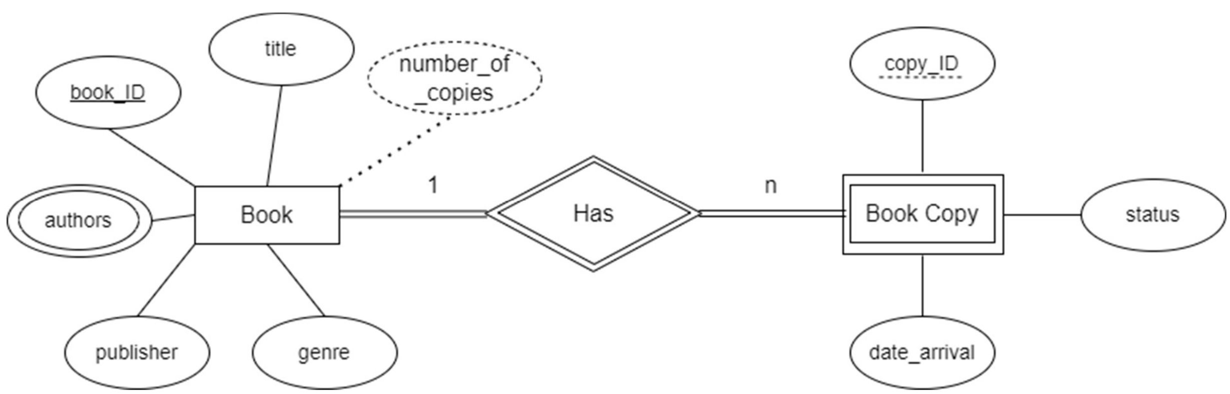


2. Please draw the ER diagram for the entity type **Reader**. [4 points]

   Answer:

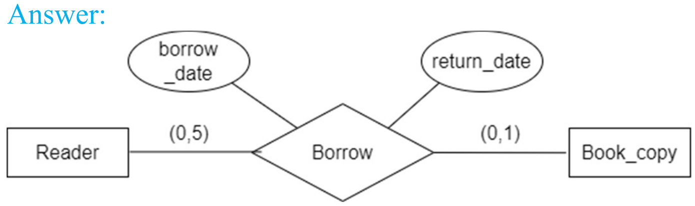


3. Suppose each book has n (n>=1) copies, with the copy ID ranging from 1 to n. Please draw the entity type **Book Copy, Book** and the relationship **Has** between them. [8 points]



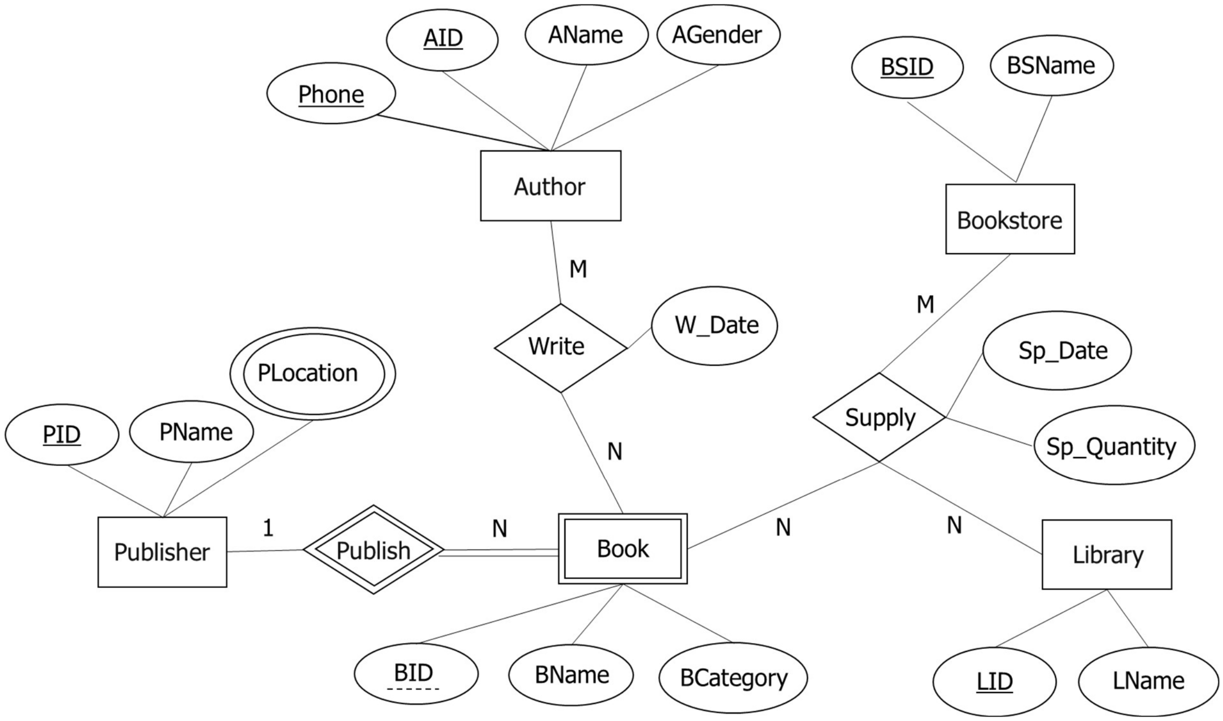
4. Suppose a reader can borrow a maximum of 5 book copies at the same time, and each book copy can be at most borrowed by only one reader at a time. Please draw the ER diagram for the relationship **Borrow** between **Reader** and **Book Copy** by using the **min-max notation**. (The attributes of both entities can be ignored.) [4 points]

Answer:

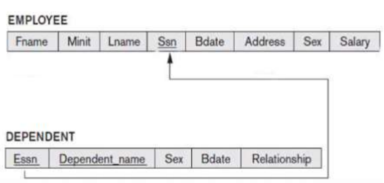


Problem TWO: Relational Model [20 points]

1. Please convert the following completed ER diagram into Relational Schema. [8 points]

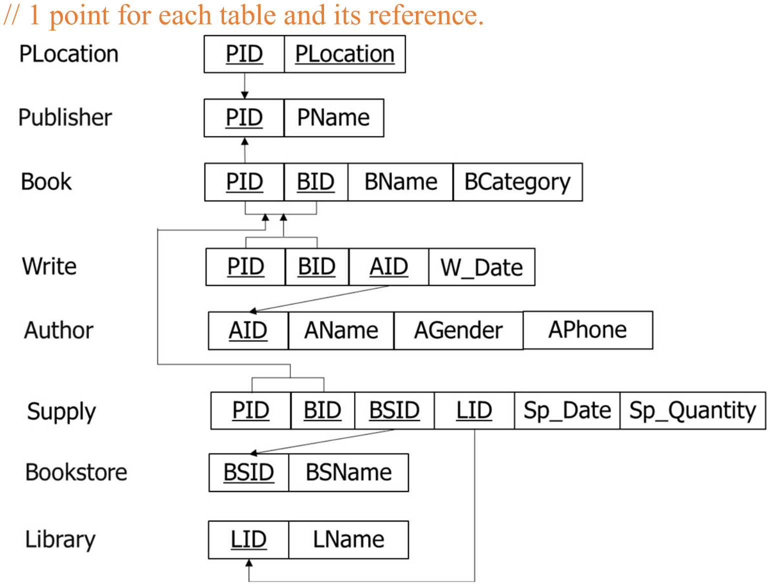


Note: you can define a relation in the sample format below:



Answer:

// 1 point for each table and its reference.



2. How many primary keys, candidate keys, and superkeys are there for the relation 'Author'? [4 points]

Answer:

It has 1 primary key, 2 candidate key and 12 superkeys

3. Assuming that the tables for the entities 'Publisher', 'Author', 'Bookstore', and 'Library' already exist, please create tables for the entity 'Book' and the relationship 'Supply', while defining the primary keys and foreign keys using SQL statements. (Hint: you can define the datatype of attributes by yourself). [8 points]

Answer:
```
CREATE TABLE Book
(
BID INT,
PID INT,
BName VARCHAR(200),
BCategory VARCHAR (50),
PRIMARY KEY(BID, PID),
FOREIGN KEY (PID) REFERENCE Publisher (PID)
);


CREATE TABLE Supply
(
BID INT,
PID INT,
BSID INT,
LID INT
Sp_Date Date,
Sp_Quantity INT,
PRIMARY KEY(BID, PID,BSID,LID),
FOREIGN KEY (PID,BID) REFERENCE Book (PID,BID),
FOREIGN KEY (BSID) REFERENCE Bookstore (BSID),
FOREIGN KEY (LID) REFERENCE Library (LID)
);
```

Problem Three: Integrity Constraints [20 points]

1 Suppose we have a relational database of University system which contains three tables Professor(Prof_id, Name, Department, Gender, Birth_data, Email), Course(Course_id, Title, Department, Prof_id) and Enrollment(Student_id, Course_id, Grade). The current state of the database is shown in the following tables. [15 points]

Professor

Prof_id	Name	Department	Gender	Birth_date	Email
P001	John Doe	Computer Science	Male	May. 5, 1990	johndoe@university.ed u
P002	Jane Smith	Mathematics	Female	Jul. 27, 2006	janesmith@university.e du
P003	Richard Roe	Physics	Male	Aug. 13, 1990	richardroe@university. edu
P004	David Johnson	Biology	Male	Dec. 31, 1998	sacraver@hotmail.com
P005	Emily Johnson	Chemistry	Female	Nov. 8, 1996	emilyjohnson@univers

					ity.edu
P006	Michael Anderson	English	Male	Feb. 17, 1993	michaelanderson@university.edu
P007	Linda White	History	Female	Aug. 13, 1995	lindawhite@university.edu
P008	David Johnson	Computer Science	Male	Apr. 30, 1990	dav.johnson@example.com
P009	Linda White	Physics	Female	Apr. 14, 1993	linw@verizon.net

Course

Course_id	Title	Department	Prof_id
C001	Introduction to Python	Computer Science	P001
C002	Advanced Mathematics	Mathematics	P002
C003	Theoretical Physics	Physics	P003
C004	General Biology	Biology	P004
C005	Organic Chemistry	Chemistry	P005
C006	Shakespearean Literature	English	P006
C007	World History	History	P007
C008	Data Structures	Computer Science	P001
C009	Calculus II	Mathematics	P002

Enrollment

Student_id	Course_id	Grade
S001	C001	A
S002	C001	B
S003	C002	A
S004	C003	C
S005	C003	B
S001	C004	A
S002	C005	B
S006	C006	A
S007	C007	A

(1) Supposing all tables are created, please use the command "Alter Table" to define all primary keys and foreign keys of all tables (Write corresponding SQL statements). [5 points]

Answer:

ALTER TABLE Professor ADD CONSTRAINT PK_prof PRIMARY KEY (Prof_id); (1 point)
ALTER TABLE Course ADD CONSTRAINT PK_course PRIMARY KEY (Course_id); (1 point)
ALTER TABLE Enrollment ADD CONSTRAINT PK_enroll PRIMARY KEY (Student_id, Course_id); (1 point)

ALTER TABLE Course ADD CONSTRAINT FK_course FOREIGN KEY (Prof_id) REFERENCES (Professor(Prof_id)); (1 point)
ALTER TABLE Enrollment ADD CONSTRAINT FK_enroll FOREIGN KEY (Course_id) REFERENCES (Course(Course_id)); (1 point)

(2) For 3.1 and 3.2 below, suppose each of the following operations is applied directly to the database. Discuss all integrity constraints violated by each operation if any, and the different ways of enforcing these constraints.

3.1) Insert < ‘S002’, ‘C001’, 73 > into Enrollment. [5 points]

Answer:

Violates both the key constraint and domain constraint. Violates key constraint because （Student_id, Course_id) is the primary key and there already exists an Enrollment tuple with Student_id =’ S002’ and Course_id=‘C001’. Violates domain constraint because the data type of Grade should be character (e.g, ‘A, B, C, D’), but 73 is an interger. (1 point)

We may enforce the key constraint by: (i) rejecting the insertion, or (ii) changing the value of （Student_id, Course_id) in the new tuple to a non-duplicate and non-null value in the new tuple. (2 points)

We may enforce the domain constraint by: (i) rejecting the insertion, or (ii) changing the value of Grade into a character (e.g, ‘A, B, C, D’) rather than number 73. (2 points)

3.2) Insert <NULL, ‘Computational Imaging’, ‘Computer Science’, ‘P012’ > into Course. [5 points]

Answer:

Violates both the entity integrity and referential integrity. Violates entity integrity because Course_id is the primary key, it does not allow null value. Violates referential integrity because foreign key Prof_id='P012' and there is no tuple in the Professor table with id='P012'. (1 point)

We may enforce entity integrity constraint by: (i) rejecting the insertion, or (ii) changing the value of Course_id in the new tuple to a non-duplicate and non-null value of Course_id in the Course table. (2 points)

We may enforce the referential by: (i) rejecting the insertion, (ii) changing the value of Prof_id to an existing id value in Professor, or (iii) inserting a new Professor tuple with Prof_id='P012'. (2 points)

2 Given a relation schema R (A,B,C,D,E) with the function dependency set F={ AB→CD， C→B, D→E, E→A}, please determine whether each of the following functional dependency is in F+. (Hint: no need to show the proof.) [5 points]

1) AC→E

2) AE→B

3) $BE \rightarrow D$

4) $BC \rightarrow A$

5) $CD \rightarrow AB$

Answer: 1), 3),5) are in the F+ but 2) 4) are not in the F+.(1 points for each FD)

Problem Four: Normalization [20 points]

1. Suppose we have a relation R with attributes A, B, C, D, E, F, G, H and the functional dependencies are: AC→B, BD→E, CE→FG, A→H. Please prove that FD: ACD→ FG holds. [5 points]

Answer:

1. AC→B (Given)
2. ACD→BD (augmentation rule on 1)
3. BD→E (Given)
4. ACD→E (transitivity rule on 2&3)
5. ACD→C (reflective rule)
6. ACD→CE (union rule on 4&5)
7. CE→FG (given)
8. ACD→FG (transitivity on 6&7)

2. Let's consider the following relation R storing the information about album retailers.
R(StoreID, StoreAddress, AlbumID, ReleaseYear, Artist, BirthPlace, BirthYear, Inventory, Price).
It has following functional dependencies:
StoreID → StoreAddress
AlbumID → {ReleaseYear, Artist}
Artist → {BirthPlace, BirthYear}
{StoreID, AlbumID} → Inventory
Inventory → Price

(1) Identify all the candidate keys in this table. [2 Points]
Answer: {StoreID, AlbumID}

(2) Is the relation R in 2NF and why? If not, decompose it into **Three** tables which satisfy 2NF but not 3NF. [5 Points]
Answer: Not in 2NF, because there exists partial function dependency on primary keys, StoreID → StoreAddress, AlbumID → {ReleaseYear, Artist, BirthPlace, BirthYear}.

R1 (<u>StoreID</u>, StoreAddress)
R2 (<u>AlbumID</u>, ReleaseYear, Artist, BirthPlace, BirthYear)
R3 (<u>StoreID</u>, <u>AlbumID</u>, Inventory, Price)

(3) Does your decomposition in (2) satisfy 3NF and why? If not, normalize it into 3NF. [5 Points]
Answer: Not in 3NF, because there exists transitive function dependency on primary keys: AlbumID

→ Artist → {BirthPlace, BirthYear}, {StoreID, AlbumID} → Inventory → Price.

R1 (StoreID, StoreAddress)
R2A (AlbumID, ReleaseYear, Artist)
R2B (Artist, BirthPlace, BirthYear)
R3A (StoreID, AlbumID, Inventory)
R3B (Inventory, Price)

(4) Does your decomposition in (3) satisfy BCNF and why? If not, normalize it into BCNF. [3 Points]
Answer: Yes, it already satisfies BCNF. Because in each table, for each functional dependency, the left-hand side is a super key.

Problem FIVE: SQL

Given the following four relations about the information of course offerings in a university. [20 points]

- Student (**StudentID**: integer, **Name**: string, **Age**: integer, **Department**: string, **GPA**:float)
  // describe the student's information including ID, name, age, GPA, and the major department the student belongs to.
- Teacher (**TeacherID**: integer, **Name**: string, **Department**: string)
  // describe the teacher's information including ID, name, and the department the teacher belongs to.
- Course (**CourseID**: integer, **Name**: string, **Department**: string, **TeacherID**: integer)
  // describe the course's information including ID, name, and the department which offers the course.
- Grade (**StudentID**: integer, **CourseID**: integer, **Score**: integer)
  // describe which student takes which course and get how many scores in that course

Suppose now we have a valid database state. Answer the following questions by completing missing parts of given SQL statement.

(1) List the StudentID of students who are majoring in '**Physics**' and have enrolled in the courses offered by the '**CS**' department that are taught by teachers from the '**Math**' Department. [5 points]

SELECT DISTINCT s.StudentID

FROM ____

WHERE ____

____;

Answer:

```
SELECT DISTINCT s.StudentID
FROM Student AS s, Grade AS g, Course AS c, Teacher AS t
WHERE s.StudentID = g.StudentID AND g.CourseID = c.CourseID AND c.TeacherID =
t.TeacherID
AND s.Department = ‘Physics’ AND c.Department = ‘CS’ AND t.Department = ‘Math’;
```

(2) List the StudentID of students who have not taken any courses outside their major department. [5 points]

```
SELECT DISTINCT s.StudentID
FROM Student AS s
WHERE _____________ (
        SELECT *
        FROM ______________________________________________
        WHERE ____________________________________________
        );
```

Answer:

```
SELECT DISTINCT s.StudentID
FROM Student AS s
WHERE NOT EXISTS (
        SELECT *
        FROM Grade AS g, Course AS c
        WHERE g.CourseID = c.CourseID AND s.StudentID = g.StudentID AND
s.Department <> c.Department
);
```

(3) The '**Algorithms**' course offered by the '**CS**' department has students from different departments. List the average scores of students of different departments and arrange the list in descending order of the scores. Only those departments with more than 5 students enrolled are included. [5 points]

```
SELECT s.Department, ___________________________________
FROM Student AS s, Grade AS g, Course AS c
WHERE _______________________________________________
GROUP BY ____________________________________________
```

HAVING ____

ORDER BY ____;

Answer:
SELECT s.Department, AVG(g.Score)
FROM Student AS s, Grade AS g, Course AS c
WHERE s.StudentID = g.StudentID AND g.courseID = c.CourseID
AND c.Name = 'Algorithms' AND c.Department = 'CS'
GROUP BY s.Department
HAVING COUNT(s.StudentID) > 5
ORDER BY AVG(g.Score) DESC;

(4) The school wants to identify exceptional students to mentor incoming freshmen. List the StudentID and Department of students who have a GPA greater than 3.6 **or** have scored above 90 in at least one course **within their major department**. [5 points]

(SELECT s.StudentID, s.Department

FROM ____

WHERE ____ **)**

____

(SELECT g.StudentID, s.Department

FROM ____

WHERE____**);**

Answer:
(SELECT s.StudentID, s.Department
FROM Student AS s
WHERE s.GPA > 3.6)
UNION
(SELECT g.StudentID, s.Department
FROM Student AS s, Grade AS g, Course AS c
WHERE s.StudentID = g.StudentID AND g.CourseID = c.CourseID AND g.Score > 90 AND c.Department = s.Department);