



Midterm solMidterm solMidterm solMidterm sol

Database Systems (City University of Hong Kong)

## Problem I: ER Model [20 points]

Consider a university database for the scheduling of classrooms for final exams. There are five entities: EXAM, ROOM, COURSE, SECTION, STUDENT.

Followings are the description of the database:

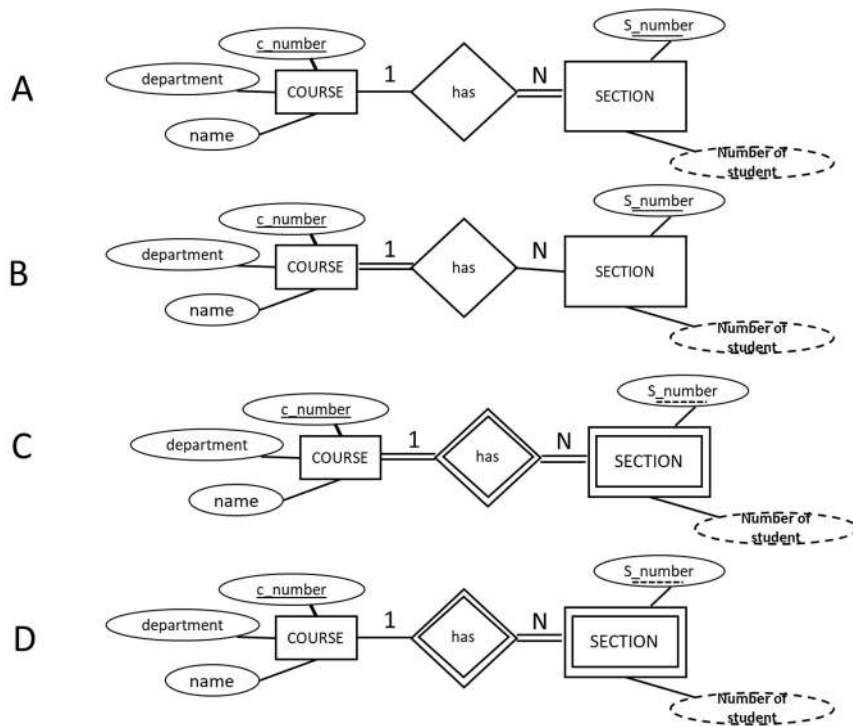
- Each EXAM has attributes exam-id (unique), starting time, end time, and invigilator.
- Each COURSE may or may not have a final exam. Each COURSE has attributes: course number (unique), department, and course name. For COURSEs offered to many students, they may consist of multiple SECTIONs. By default, each COURSE has at least one SECTION.
- SECTION is a weak-entity type. It is dependent on the COURSE. Each section has two attributes: section number and number of students. Each section has at least one student enrolled in.
- STUDENT has attributes student id (unique), name, and age. A student can be enrolled in no or several course sections.
- Each ROOM has attributes room number, building, and capacity. One room may serve as the venue of multiple exams. One exam must be held in one room.

(1) What is the definition of weak entity type? How to uniquely identify a weak-entity instance in a database? [4 points]

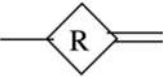
**Sol:** Weak entity is the entity types that do not have key attributes of their own. (2 points)

A weak entity instance is identified by the combination of its owner's primary key and its partial key. (2 points)

(2) To represent the relationship between SECTION and COURSE, which one of the following is correct? [4 points]

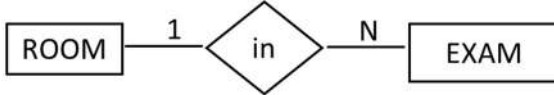
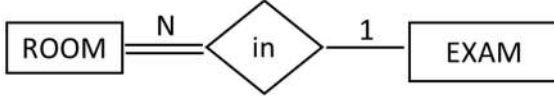
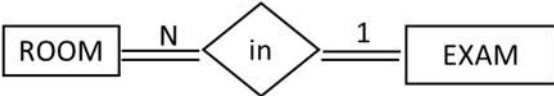
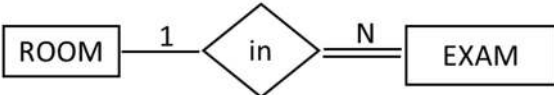


Sol:C

(3) In the diagram , what do double line and single line stand for, respectively? [4 points]

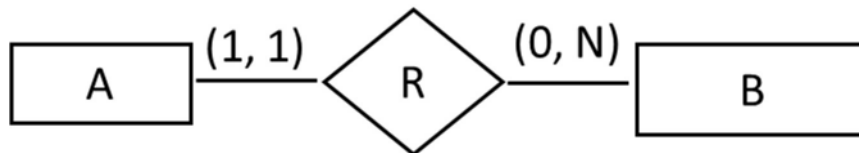
Sol: Double line: total participation/ mandatory participation (2 points)  
Single line: partial participation (2 points)

(4) To represent the relationship between ROOM and EXAM, which one of the following is correct? [4 points]

- A 
- B 
- C 
- D 
- E None

Sol: D

(5) For the relationship “enroll” between STUDENT and SECTION, please represent its cardinality and participation constraints with the min-max notation. [4 points]  
(Note: You do not need to draw a diagram. Please represent min-max notation as the example shown below.)



You can write the min-max notation of this relationship in the following format:

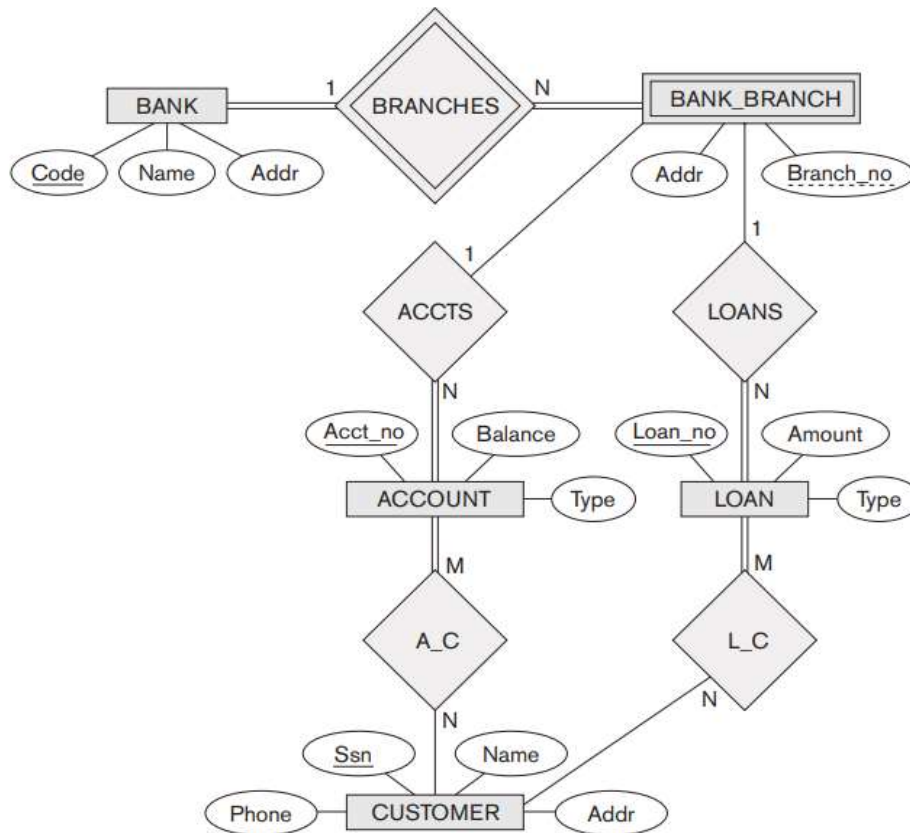
[A, R, B] : (1, 1)  
[B, R, A] : (0, N)

Sol:

[STUDENT, enroll, SECTION]: (0, N) (2 points)  
[SECTION, enroll, STUDENT]: (1, N) (2 points)

## Problem II: Relational Model [19 points]

(1) Consider the ER diagram shown in the figure below for part of a BANK database. Each bank can have multiple branches, and each branch can have multiple accounts and loans. Please convert this ER diagram into relational model. [16 points]



Note: please define a relation in this format: **Tablename (attribute1, attribute2, attribute3,... )** and indicate the reference of foreign key by an arrow in this way: **TableA.attribute1 -> TableB.attribute2**, which means the foreign key TableA.attribute1 references TableB.attribute2.

Sol:

//1 point for each line

BANK (Code, name, address)

BANK\_BRANCH(Code, Branch\_no, Addr)

ACCOUNT(Acc\_no, Balance, Type, Code, Branch\_no)

LOAN (Loan\_no, Amount, Type, Code, Branch\_no)

CUSTOMER (Ssn, Name, Addr, Phone)

A\_C (Ssn, Acc\_no)

L\_C (Ssn, Loan\_no)

BANK\_BRANCH.Code -> BANK.Code

ACCOUNT.Code -> BANK\_BRANCH.Code  
 ACCOUNT.Branch\_no -> BANK\_BRANCH.Branch\_no  
 LOAN.Code -> BANK\_BRANCH.Code  
 LOAN.Branch\_no -> BANK\_BRANCH.Branch\_no  
 A\_C.Ssn -> CUSTOMER.Ssn  
 A\_C.Acc\_no -> ACCOUNT.Acc\_no  
 L\_C.Ssn -> CUSTOMER.Ssn  
 L\_C.Loan\_no -> LOAN.Loan\_no

(2) When converting a one-to-one relationship into a foreign key, please explain why including the foreign key to the total participation side is better than including the foreign key to the partial participation side? [3 points]

Sol:

Because it can avoid null value in the foreign key attribute.

### Problem III: Integrity Constraints [15 Points]

Suppose we have a relational database containing three tables:

Student(sid, name, sex, enroll\_date, depart\_id)

Department(depart\_id, depart\_name, subject\_num, depart\_lead\_name)

Professor(professor\_id, professor\_name, sex, salary, research\_interest, depart\_id)

The current state of the database is shown in the following tables:

**Student**

sid	name	enroll_date	sex	depart_id
10000	Hayley Santos	2020-09-01	F	1
10001	Melanie Jensen	2019-12-15	M	3
10002	Caroline Merritt	2018-09-01	M	5
10003	Rosie Lambert	2018-09-21	M	4
10004	Kelly Kapoor	2021-12-21	F	3
10005	Kitty Peck	2020-09-01	F	2
10006	Susan McKay	2018-09-01	F	2
10007	Rosie Lambert	2019-09-01	M	1

**Department**

depart_id	depart_name	subject_num	depart_lead_name
1	Computer Science	15	Karen Henderson

2	Math	23	Hermione Rosario
3	Arts	32	Lara Lynch
4	Geography	11	Emelia Bell
5	Sports	15	Aidan Contreras

### Professor

professor_id	professor_name	sex	salary	research_interest	depart_id
200	Eden Riggs	F	500,000	Basketball	5
201	Carys Roy	M	700,000	Mathematics	2
202	Sofia Perry	F	450,000	Earth	4
203	Taylor Leon	F	550,000	Drama	3
204	Vanessa Myers	M	600,000	GAN	1
205	Jessie Ramsey	M	600,000	English	5
206	Taylor Leon	F	800,000	Mathematics	2

For question (2) and (3), suppose each of the following Update 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.

(1) Analyze the primary key and the foreign keys of the providing relations. [5 points]

Answer:

Primary key: [3 points]

sid is the primary key of Student.

depart\_id is the primary key of Department.

professor\_id is the primary key of Professor.

Foreign key: [2 points]

The attribute depart\_id of relation Student and relation Professor that reference relation Department.

(2) Insert the tuple <202, 'Lura David', 'F', '500,000', 'AuotML', null> into the Professor relation. [5 points]

Answer: Violates key constraint. [1 points]

Reason: Violates the key constraint because there already exists a Professor tuple with professor\_id=202. [2 points]

Solution: [2 points]

(i) Rejecting the insertion

(ii) Changing the value of professor\_id to a value that doesn't exist in Professor.

(3) Delete the Department tuple with depart\_id=2. [5 Points]

Answer: Violates referential integrity constraint [1 points]

Reason: Violates the referential integrity constraint because 2 tuples exist in the Student relations and 2 tuples exist in the Professor relations the reference the tuple being deleted from Department. [2 points]

Solution: [2points]

(i) Rejecting the deletion

(i) Delete all the tuples in the Student and Professor relations whose value for depart\_id=2.

### Problem IV Normalization [22 Points]

Consider relation schema:  $R(A,B,C,D,E,F,G)$ , and the corresponding set of functional dependencies:  $F = \{AB \rightarrow E, A \rightarrow B, B \rightarrow C, C \rightarrow D\}$ .

(1) Please write the **closure** of  $\{A\}$  with respect to the **F**. Is  $\{A\}$  the **candidate key** of  $R$ ? if not, write the candidate key of  $R$ . [4 points]

**Answer:**

$\{A\}^+ = \{A,B,C,D,E\}$ ;

$\{A\}$  is not the candidate key of  $R$ ;

$\{A,F,G\}$  is the candidate key of  $R$  since  $\{A,F,G\}^+ = \{A,B,C,D,E\}$ ; and no subset can be a superkey.

(2) Normalization is the procedure to decompose a relation into several related relations. Normalization usually requires two properties:

(I) **Non-additive** (also known as **Lossless join**), and

(II) **Preservation of the functional dependencies**.

Please explain the meaning of these two properties. [4 points]

**Answer:**

(2.1) Decomposition is reversible and no information is lost. No spurious tuple should be generated by doing a natural join of any relations; (2 marks)

(2.2) Ensure each functional dependency is represented in some individual relation or could be inferred from other dependencies after decomposition. (2 marks)

(3) Does the relation schema **R** satisfy **2NF** and why? [4 points]

**Answer:**

$R$  does not satisfy 2NF, (2 mark)

because there exists non-prime attribute partially depends on primary keys:  $A \rightarrow B$ . (2mark)

(4) If we decompose the relation schema **R** into:  $R_1(A,B,E)$ ,  $R_2(B,C,D)$ ,  $R_3(A,F,G)$ , what is the **highest norm form** that the database schema  $\{R_1, R_2, R_3\}$  satisfies and explain why? [4 points]

**Answer:**

$\{R_1, R_2, R_3\}$  satisfies 2NF, (2 mark)

Because there is no partial dependency on primary keys but exists transitive function dependency on primary keys:  $B \rightarrow C$ ,  $C \rightarrow D$ . (2 mark)



(5) Normalize **R** into **3NF** and indicate whether your solution is **BCNF**. [6 points]

**Answer:**

Solution 1:

R1(A,B,E),  
R2(B,C),  
R3(C,D),  
R4(A,F,G). (4 marks)  
Satisfies BCNF. (2 marks)

Solution 2:

R1(A,B),  
R2(A,E)  
R3(B,C),  
R4(C,D),  
R5(A,F,G). (4 marks)  
Satisfies BCNF. (2 marks)

Remarks:

Solution 1 is a lossless join and preserves the functional dependencies.

Solution 2 is a lossless join but does not preserve the functional dependencies.

## Problem V SQL [24 Points]

Given the following relations about the information of courses in a university.

Student (StudentID: integer, StudentName: string, Age: integer, Gender: string, DepartmentName: string)

Course (CourseID: integer, CourseName: string, TeacherID: integer, DepartmentName:string, Semester: string, Year: string)

Teacher (TeacherID: integer, TeacherName: string, DepartmentName:string)

Grade (StudentID: integer, CourseID: integer, Score: integer)

Department (DepartmentName: string, Target: string)

The foreign key references are listed here:

Student. DepartmentName -> Department.DepartmentName;

Course. DepartmentName -> Department.DepartmentName;

Course. TeacherID -> Teacher.TeacherID;

Teacher. DepartmentName -> Department.DepartmentName;

Grade. StudentID -> Student. StudentID

Grade. CourseID -> Course. CourseID

Suppose now we have a valid database state. Write the following queries in **SQL**.

(1) Query the ID of courses which are not taught by teachers from the same department as the course. [6 points]

Answer:

```
SELECT DISTINCT CourseID
FROM Course
WHERE NOT EXISTS ( SELECT *
                   FROM Course, Teacher
                   WHERE Course.DepartmentName = Teacher.DepartmentName );
```

(2) For teachers who taught more than two courses, retrieve teacher ID and the number of course taught by the teachers. [6 points]

Answer:

```
SELECT TeacherID, COUNT(*)
FROM Course, Teacher
WHERE Course.TeacherID = Teacher.TeacherID
GROUP BY TeacherID
HAVING COUNT(*) > 2;
```

(3) Find all course ID taught in the Fall 2021 semester, or in the Spring 2022 semester, or in both. [6 points]

Answer:

```
(SELECT DISTINCT CourseID
FROM Course
WHERE Course.Semester = 'Fall' AND Course.Year = 2021 )
UNION
(SELECT DISTINCT CourseID
FROM Course
WHERE Course.Semester = 'Spring' AND Course.Year = 2022);
```

(4) Count the average score of each of the courses from 'Computer Science' department. Order the results in the descending order of course's ID. [6 points]

Answer:

```
SELECT Course.CourseID, AVG(Score)
FROM Grade, Course,
WHERE Grade.CourseID= Course.CourseID AND Course.DepartmentName = 'Computer Science'
GROUP BY Course.CourseID;
ORDER BY Course.CourseID DESC;
```