資料庫管理 HW02

B12508026 戴偉璿

1. (a) i. **TRUE**, because `DEAN` is a **relation** between `COLLEGE` and `INSTRUCTOR`; and `CHAIR` is a **relation** between `DEPT` and `INSTRUCTOR`.

    ii. **FALSE**, there's no further restriction on `DEAN` and `CHAIR`, so one `INSTRUCTOR` can be a `CHAIR` and a `DEAN` at the same time.

    iii. **TRUE**, the relation between `STUDENT` and `HAS` is a **(0, 1)** relation, so one `STUDENT` can `HAS` zero or one `DEPT`.

    iv. **TRUE**, the cardinality between `STUDENT` and `TAKES` is **(0, N)**, so one student may take zero or more sections; while the cardinality between `SECTION` and `TAKES` is **(5, N)**, so one section must be taken by five or more students.

    v. **TRUE**, the cardinality between `COURSE` and `SECTION` is **(1, 1)**, so one section must be related to exactly one course.
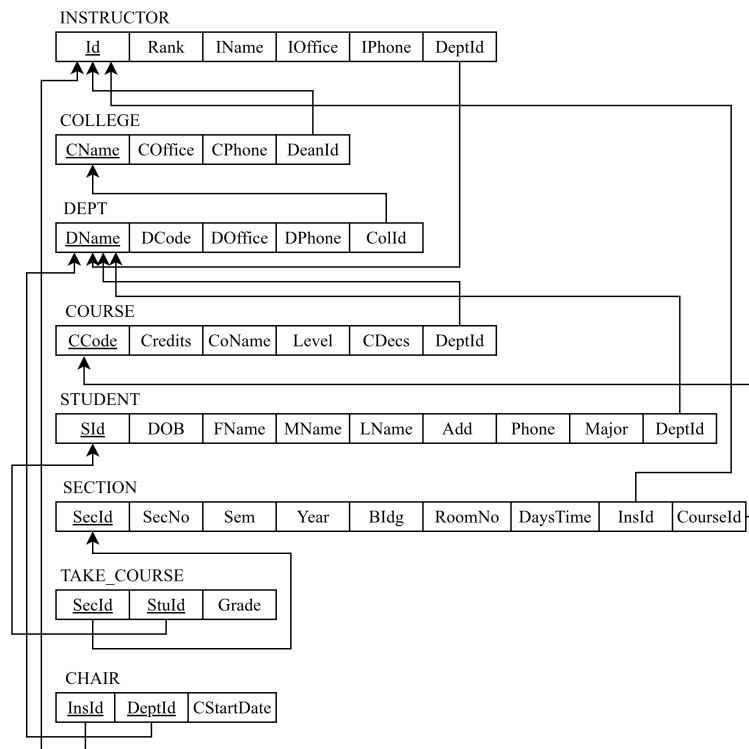
   (b) As the Figure 1 shows:

Figure 1: Relational Schema Diagram

2. (a) To record the full history of students' take or drop sections, we can just add more attributes to the `TAKES` relation. But it may cause some problems while querying the final result and grade (user must find the last record of the log to reach). So I decide to create a new weak entity to record the operation log.

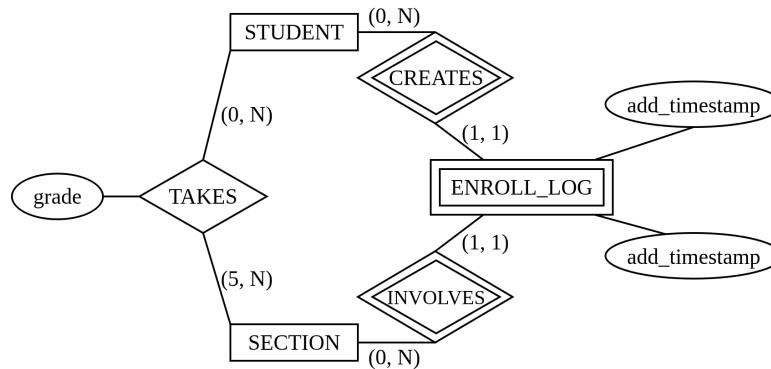If the operation is add, the `drop_data` would be record as NULL; vice versa.

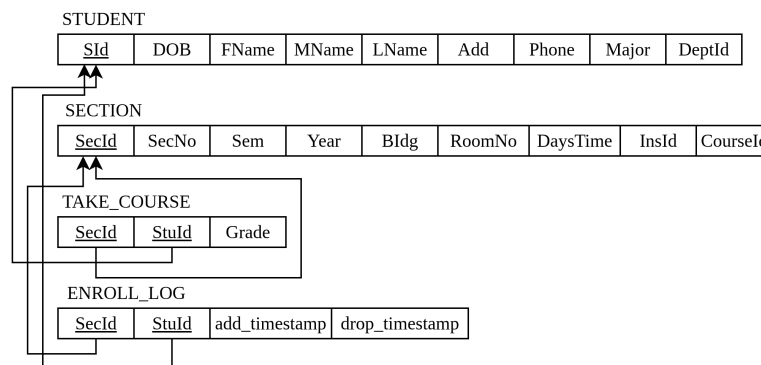Figure 2: ER Diagram with Operation Log

(b) As the Figure 3 shows:

Figure 3: Relational Schema Diagram with Operation Log

3. (a)
```
1  SELECT br.Card_no, br.Name, COUNT(*) AS LoanRec
2  FROM BOOK_LOANS bl
3  JOIN BORROWER br ON bl.Card_no = br.Card_no
4  WHERE bl.Branch_id = '[ASSIGNED_BRANCH_ID]'
5  GROUP BY br.Card_no, br.Name
6  ORDER BY LoanRec DESC;
```

(b)
```
1  SELECT lib.Branch_id, lib.Branch_name, COUNT(*) AS LoanRec
2  FROM BOOK_LOANS bl
3  JOIN LIBRARY_BRANCH lib ON lib.Branch_id = bl.Branch_id
4  WHERE bl.Date_out BETWEEN '2024-01-01' AND '2024-12-31'
5  GROUP BY lib.Branch_id, lib.Branch_name
6  ORDER BY LoanRec DESC;
```

(c)
```
1    SELECT bk.Book_id, bk.Title, COUNT(DISTINCT ba.Author_name) AS AuthorNum,
     ↪  bc.No_of_copies, COUNT(DISTINCT bl.Loan_id) AS LoanRec
2    FROM BOOK_LOANS bl
3    JOIN BOOK bk ON bl.Book_id = bk.Book_id
4    JOIN BOOK_AUTHORS ba ON bl.Book_id = ba.Book_id
5    JOIN BOOK_COPIES bc ON bl.BOOK_id = bc.BOOK_id AND bl.Branch_id = bc.Branch_id
6    WHERE bl.Branch_id = (
7        SELECT bl2.Branch_id
8        FROM BOOK_LOANS bl2
9        JOIN LIBRARY_BRANCH lib ON lib.Branch_id = bl2.Branch_id
10       WHERE bl2.Date_out BETWEEN '2024-01-01' AND '2024-12-31'
11       GROUP BY bl2.Branch_id
12       ORDER BY COUNT(*) DESC
13       LIMIT 1
14   )
15   AND bl.Date_out BETWEEN '2024-01-01' AND '2024-12-31'
16   GROUP BY bk.Book_id, bk.Title, bc.No_of_copies
17   ORDER BY LoanRec DESC;
```

(d)
```
1    SELECT bk.Book_id, bk.Title, lib.Branch_name, bc.No_of_copies
2    FROM BOOK bk
3    JOIN BOOK_COPIES bc ON bk.Book_id = bc.Book_id
4    JOIN LIBRARY_BRANCH lib ON bc.Branch_id = lib.Branch_id
5    WHERE bk.Book_id IN (
6        SELECT ba.Book_id
7        FROM BOOK_AUTHORS ba
8        GROUP BY ba.Book_id
9        HAVING COUNT(DISTINCT ba.Author_name) = 1
10   );
```

(e)
```
1    --Add new column to the DB
2    ALTER TABLE BOOK_LOANS
3    ADD COLUMN Date_return DATE;
4
5    --Execute Query
6    SELECT bk.Title, lib.Branch_name, bc.No_of_copies - COUNT(bl.Loan_id) AS AvaiCopies
7    FROM BOOK bk
8    JOIN BOOK_COPIES bc ON bk.Book_id = bc.Book_id
9    JOIN LIBRARY_BRANCH lib ON bc.Branch_id = lib.Branch_id
10   LEFT JOIN BOOK_LOANS bl ON bk.Book_id = bl.Book_id AND bc.Branch_id = bl.Branch_id AND
     ↪  bl.Date_return IS NULL
11   GROUP BY bk.Title, lib.Branch_name, bc.No_of_copies;
```

4. (a) As the Table 1 shows:

Name	Data Type	Key	Constraint	Domain
BOOK				
Book_id	varchar(15)	PK	Not Null, Unique	
Title	varchar(100)		Not Null	
Publisher_name	varchar(50)	FK → PUB-LISHER(Name)	Not Null	
BOOK_AUTHORS				
Book_id	varchar(15)	PK, FK → BOOK(Book_id)	Not Null	
Author_name	varchar(50)	PK	Not Null	
PUBLISHER				
Name	varchar(50)	PK	Not Null, Unique	
Address	varchar(100)			
Phone	varchar(15)			
BOOK_COPIES				
Book_id	varchar(15)	PK, FK → BOOK(Book_id)	Not Null	
Branch_id	varchar(10)	PK, FK → LI-BRARY_BRANCH (Branch_id)	Not Null	
No_of_copies	int		Not Null, CHECK > 0	$\{1, 2, \dots\}$
BOOK_LOANS				
Book_id	varchar(15)	PK, FK → BOOK(Book_id)	Not Null	
Branch_id	varchar(10)	PK, FK → LI-BRARY_BRANCH (Branch_id)	Not Null	
Card_no	varchar(10)	PK, FK → BOR-ROWER(Card_no)	Not Null	
Date_out	date	PK	Not Null	
Due_date	date		Not Null	
Date_return	date			
LIBRARY_BRANCH				
Branch_id	varchar(10)	PK	Not Null, Unique	
Branch_name	varchar(50)		Not Null	
Address	varchar(100)		Not Null, Unique	
BORROWER				
Card_no	varchar(10)	PK	Not Null, Unique	
Name	varchar(50)		Not Null	
Address	varchar(100)			
Phone	varchar(15)			

Table 1: Data Dictionary for Library Management System

(b)

```sql
1    CREATE TABLE PUBLISHER (
2        Name VARCHAR(50) PRIMARY KEY,
3        Address VARCHAR(100),
4        Phone VARCHAR(15)
5    );
6
7    CREATE TABLE LIBRARY_BRANCH (
8        Branch_id VARCHAR(10) PRIMARY KEY,
9        Branch_name VARCHAR(50) NOT NULL,
10       Address VARCHAR(100) NOT NULL UNIQUE
11   );
12
13   CREATE TABLE BORROWER (
14       Card_no VARCHAR(10) PRIMARY KEY,
15       Name VARCHAR(50) NOT NULL,
16       Address VARCHAR(100),
17       Phone VARCHAR(15)
18   );
19
20   CREATE TABLE BOOK (
21       Book_id VARCHAR(15) PRIMARY KEY,
22       Title VARCHAR(100) NOT NULL,
23       Publisher_name VARCHAR(50) NOT NULL,
24       FOREIGN KEY (Publisher_name) REFERENCES PUBLISHER(Name)
25   );
26
27   CREATE TABLE BOOK_AUTHORS (
28       Book_id VARCHAR(15) NOT NULL,
29       Author_name VARCHAR(50) NOT NULL,
30       PRIMARY KEY (Book_id, Author_name),
31       FOREIGN KEY (Book_id) REFERENCES BOOK(Book_id)
32   );
33
34   CREATE TABLE BOOK_COPIES (
35       Book_id VARCHAR(15) NOT NULL,
36       Branch_id VARCHAR(10) NOT NULL,
37       No_of_copies INT NOT NULL CHECK (No_of_copies > 0),
38       PRIMARY KEY (Book_id, Branch_id),
39       FOREIGN KEY (Book_id) REFERENCES BOOK(Book_id),
40       FOREIGN KEY (Branch_id) REFERENCES LIBRARY_BRANCH(Branch_id)
41   );
42
43   CREATE TABLE BOOK_LOANS (
44       Book_id VARCHAR(15) NOT NULL,
45       Branch_id VARCHAR(10) NOT NULL,
46       Card_no VARCHAR(10) NOT NULL,
47       Date_out DATE NOT NULL,
48       Due_date DATE NOT NULL,
49       Date_return DATE,    -- can be NULL
50       PRIMARY KEY (Book_id, Branch_id, Card_no, Date_out),
51       FOREIGN KEY (Book_id) REFERENCES BOOK(Book_id),
52       FOREIGN KEY (Branch_id) REFERENCES LIBRARY_BRANCH(Branch_id),
53       FOREIGN KEY (Card_no) REFERENCES BORROWER(Card_no)
54   );
```