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 following diagram:

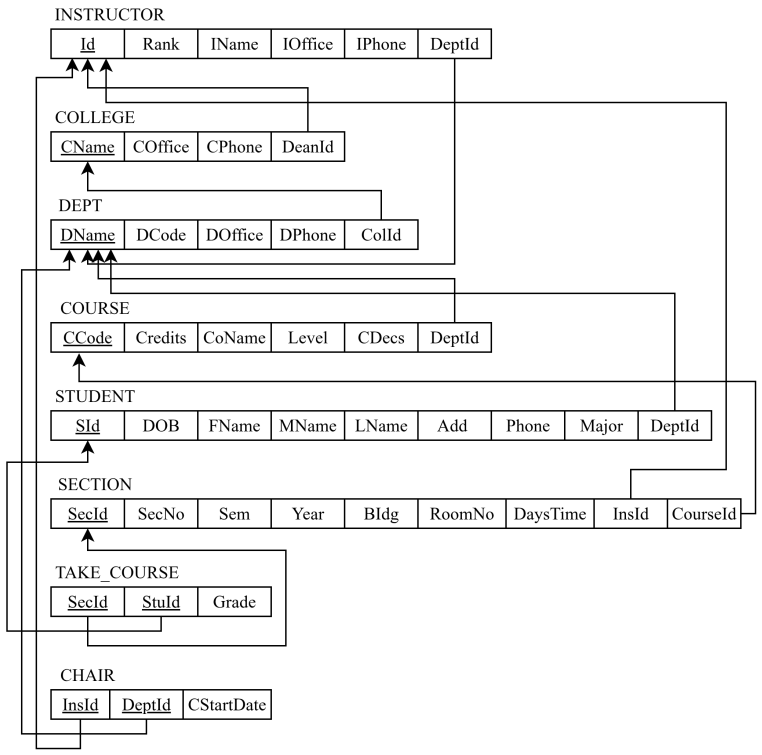

Figure 1: Relational Schema Diagram

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.

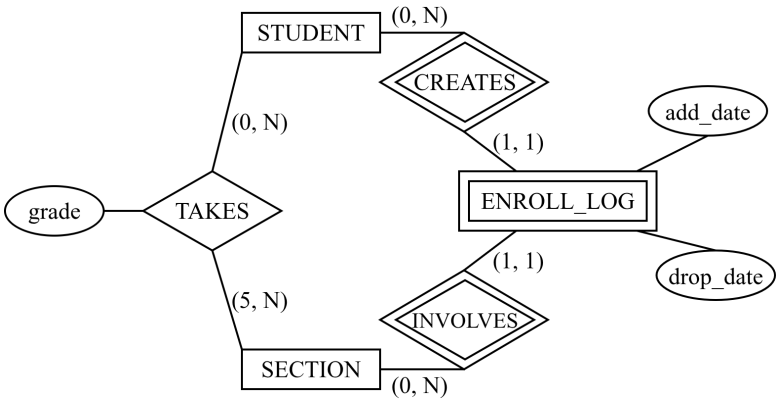

Figure 2: ER Diagram with Operation Log

(b) As the following diagram:

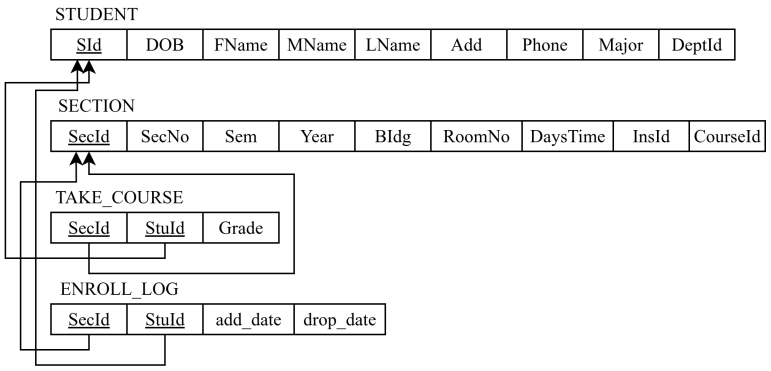

Figure 3: Relational Schema Diagram with Advisor

3. (a)

```sql
SELECT br.Card_no, br.Name, COUNT(*) AS LoanRec
FROM BOOK_LOANS bl
JOIN BORROWER br ON bl.Card_no = br.Card_no
WHERE bl.Branch_id = '[ASSIGNED_BRANCH_ID]'
GROUP BY br.Card_no, br.Name
ORDER BY LoanRec DESC;
```

(b)

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

(c)

```sql
SELECT bk.Book_id, bk.Title, COUNT(DISTINCT ba.Author_id) AS AuthorNum, bc.No_of_copies,
↪  COUNT(DISTINCT bl.Loan_id) AS LoanRec
FROM BOOK_LOANS bl
JOIN BOOK bk ON bl.Book_id = bk.Book_id
JOIN BOOK_AUTHORS ba ON bl.Book_id = ba.Book_id
JOIN BOOK_COPIES bc ON bl.BOOK_id = bc.BOOK_id AND bl.Branch_id = bc.Branch_id
WHERE bl.Branch_id = (
    SELECT bl2.Branch_id
    FROM BOOK_LOANS bl2
    JOIN LIBRARY_BRANCH lib ON lib.Branch_id = bl2.Branch_id
    WHERE bl2.Date_out BETWEEN '2024-01-01' AND '2024-12-31'
    GROUP BY bl2.Branch_id
    ORDER BY COUNT(*) DESC
    LIMIT 1
)
AND bl.Date_out BETWEEN '2024-01-01' AND '2024-12-31'
GROUP BY bk.Book_id, bk.Title, bc.No_of_copies
ORDER BY LoanRec DESC;
```

(d)

```sql
SELECT bk.Book_id, bk.Title, lib.Branch_name, bc.No_of_copies
FROM BOOK bk
JOIN BOOK_COPIES bc ON bk.Book_id = bc.Book_id
JOIN LIBRARY_BRANCH lib ON bc.Branch_id = lib.Branch_id
WHERE bk.Book_id IN (
    SELECT ba.Book_id
    FROM BOOK_AUTHOR ba
    GROUP BY ba.Book_id
    HAVING COUNT(DISTINCT ba.Author_id) = 1
);
```

(e)

```sql
--Add new column to the DB
ALTER TABLE BOOK_LOANS
ADD COLUMN Date_return DATE;

--Execute Query
SELECT bk.Title, lib.Branch_name, bc.No_of_copies - COUNT(bl.Loan_id) AS AvaiCopies
FROM BOOK bk
JOIN BOOK_COPIES bc ON bk.Book_id = bc.Book_id
JOIN LIBRARY_BRANCH lib ON bc.Branch_id = lib.Branch_id
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
GROUP BY bk.Title, lib.Branch_name, bc.No_of_copies
```

4. (a) As the following tables:

BOOK

Column Name	Data Type	Key	Constraint	Domain
Book_id	varchar(15)	PK	Not Null, Unique	
Title	varchar(100)		Not Null	
Publisher_name	varchar(50)	FK → PUBLISHER(Name)	Not Null	

BOOK_AUTHORS

Column Name	Data Type	Key	Constraint	Domain
Book_id	varchar(15)	PK, FK → BOOK(Book_id)	Not Null	
Author_name	varchar(50)	PK	Not Null	

PUBLISHER

Column Name	Data Type	Key	Constraint	Domain
Name	varchar(50)	PK	Not Null, Unique	
Address	varchar(100)			
Phone	varchar(15)			

BOOK_COPIES

Column Name	Data Type	Key	Constraint	Domain
Book_id	varchar(15)	PK, FK → BOOK(Book_id)	Not Null	
Branch_id	varchar(10)	PK, FK → LIBRARY_BRANCH(Branch_id)	Not Null	
No_of_copies	int		Not Null, CHECK ≥ 0	$\{0, 1, 2, \dots\}$

BOOK_LOANS

Column Name	Data Type	Key	Constraint	Domain
Book_id	varchar(15)	PK, FK → BOOK(Book_id)	Not Null	
Branch_id	varchar(10)	PK, FK → LIBRARY_BRANCH(Branch_id)	Not Null	
Card_no	varchar(10)	PK, FK → BORROWER(Card_no)	Not Null	
Date_out	date	PK	Not Null	
Due_date	date		Not Null	
Date_return	date		可為 Null	

LIBRARY_BRANCH

Column Name	Data Type	Key	Constraint	Domain
Branch_id	varchar(10)	PK	Not Null, Unique	
Branch_name	varchar(50)		Not Null	
Address	varchar(100)			

BORROWER

Column Name	Data Type	Key	Constraint	Domain
Card_no	varchar(10)	PK	Not Null, Unique	
Name	varchar(50)		Not Null	
Address	varchar(100)			
Phone	varchar(15)			