

資料庫系統_第七組

2024-06-17

CONTENTS

- 第一題：3NF
- 第二題：ER-Diagram
- 第三題：SQL解決問題
- 第四題：資料庫實作資料庫正規化



第一題

3NF

1NF

欄位名稱							
semester	1112	1112	1112	1112	1112	1112	1112
course_no	A0001	A0002	A0002	A0003	A0003	A0003	A0003
course_name	微積分	計算機概論	計算機概論	統計學習	統計學習	統計學習	統計學習
course_type	必修	必修	必修	選修	選修	選修	選修
course_room	K205	L102	L102	M-605	M-605	M-605	M-605
course_building	工程一館	工程五館	工程五館	鴻經館	鴻經館	鴻經館	鴻經館
course_time	—567	二34,五4	二34,五4	四567	四567	四567	四567
course_credit	2	3	3	3	3	3	3
course_limit	100	60	60	55	55	55	55
course_status	開課	開課	開課	開課	開課	開課	開課
curriculum_field	理論數學	人工智慧,資料科學與多媒體	資料科學與多媒體	財務工程,統計推論	財務工程	統計推論	統計推論
teacher_name	岳飛	陸羽	陸羽	劉邦,項羽	項羽	劉邦,項羽	項羽
student_name	張飛	關羽	關羽	劉備	劉備	劉備	劉備
student_dept	數學系	資訊工程系	資訊工程系	資訊管理系	資訊管理系	資訊管理系	資訊管理系
student_grade	1	1	1	1	1	1	1
student_status	在學	在學	在學	在學	在學	在學	在學
student_class	A	B	B	A	A	A	A
select_result	中選	中選	中選	落選	落選	落選	落選
course_score	90	63	63	-	-	-	-
feedback_rank	5	2	2	-	-	-	-

2NF

欄位名稱				
course_no	A0001	A0002	A0003	
course_name	微積分	計算機概論	統計學習	
course_type	必修	必修	選修	
course_credit	2	3	3	
欄位名稱				
class_grade_id	CG0001	CG0002	CG0003	
student_dept	數學系	資訊工程系	資訊管理系	
student_grade	1	1	1	
student_class	A	B	A	
欄位名稱				
student_id	S0001	S0002	S0003	
student_name	張飛	關羽	劉備	
student_status	在學	在學	在學	
class_grade_id	CG0001	CG0002	CG0003	

欄位名稱				
courses_id	C0001	C0002	C0003	C0003
teacher_id	T0001	T0002	T0003	T0004

欄位名稱					
course_no	A0001	A0002	A0002	A0003	A0003
curriculum_field	理論數學	人工智慧	資料科學與多媒體	財務工程	統計推論

欄位名稱				
courses_id	C0001	C0002	C0003	
semester	1112	1112	1112	
course_no	A0001	A0002	A0003	
course_room	K205	L102	M-605	
course_building	工程一館	工程五館	鴻經館	
course_time	—567	—34,五4	四567	
course_limit	100	60	55	
course_status	開課	開課	開課	

欄位名稱				
courses_id	C0001	C0002	C0003	
student_id	S0001	S0002	S0003	
select_result	中選	中選	落選	
course_score	90	63	-	
feedback_rank	5	2	-	

欄位名稱				
teacher_id	T0001	T0002	T0003	T0004
teacher_name	岳飛	陸羽	劉邦	項羽

3NF

COURSE

欄位名稱	資料型態	相依資訊	欄位說明			
course_no	varchar(10)	主鍵	課程編號	A0001	A0002	A0003
course_name	varchar(255)		課程名稱	微積分	計算機概論	統計學習
course_type	varchar(10)		選修別 (必修 / 或選修)	必修	必修	選修
course_credit	INTEGER		學分數	2	3	3

COURSE_FIELD

欄位名稱	資料型態	相依資訊	欄位說明					
course_no	varchar(10)	主鍵；外來鍵	課程編號	A0001	A0002	A0002	A0003	A0003
curriculum_field	TEXT	主鍵	課程領域	理論數學	人工智慧	資料科學與多媒體	財務工程	統計推論

SEMESTER_COURSES

欄位名稱	資料型態	相依資訊	欄位說明			
courses_id	varchar(10)	主鍵	當學期課程編號	C0001	C0002	C0003
semester	varchar(4)		學期別 (1112 則為 111 下學期)	1112	1112	1112
course_no	varchar(10)	外來鍵	課程編號	A0001	A0002	A0003
course_room	varchar(20)	外來鍵	授課教室	K205	L102	M-605
course_time	varchar(20)		授課時間 (一 123 : 表示週一早上 1-3 節) · 一門課有多節次則以逗點隔開	一 567	二 34,54	四 567
course_limit	INTEGER		課程人數限制	100	60	55
course_status	varchar(10)		課程狀態 (開課 / 停開)	開課	開課	開課

3NF

LOCATION

欄位名稱	資料型態	相依資訊	欄位說明				
course_room	varchar(20)	主鍵	授課教室	K205	L102	M-605	
course_building	varchar(20)		授課地點	工程一館	工程五館	鴻經館	

LECTURING

欄位名稱	資料型態	相依資訊	欄位說明				
courses_id	varchar(10)	主鍵；外來鍵	當學期課程編號	C0001	C0002	C0003	C0003
teacher_id	varchar(10)	主鍵；外來鍵		T0001	T0002	T0003	T0004

TEACHER

欄位名稱	資料型態	相依資訊	欄位說明				
teacher_id	varchar(10)	主鍵	授課教師編號	T0001	T0002	T0003	T0004
teacher_name	varchar(20)		授課教師姓名	岳飛	陸羽	劉邦	項羽

STUDENT

欄位名稱	資料型態	相依資訊	欄位說明				
student_id	varchar(10)	主鍵	學生編號	S0001	S0002	S0003	
student_name	varchar(20)		學生姓名	張飛	關羽	劉備	
student_status	varchar(10)		學生在學狀態 (在學 / 休學 / 退學)	在學	在學	在學	
class_grade_id	varchar(10)	外來鍵	系級編號	CG0001	CG0002	CG0003	

3NF

CLASS_GRADE

欄位名稱	資料型態	相依資訊	欄位說明			
class_grade_id	varchar(10)	主鍵	系級編號	CG0001	CG0002	CG0003
student_dept	varchar(30)		學生系所	數學系	資訊工程系	資訊管理系
student_grade	INTEGER		學生年級	1	1	1
student_class	varchar(1)		學生班別	A	B	A

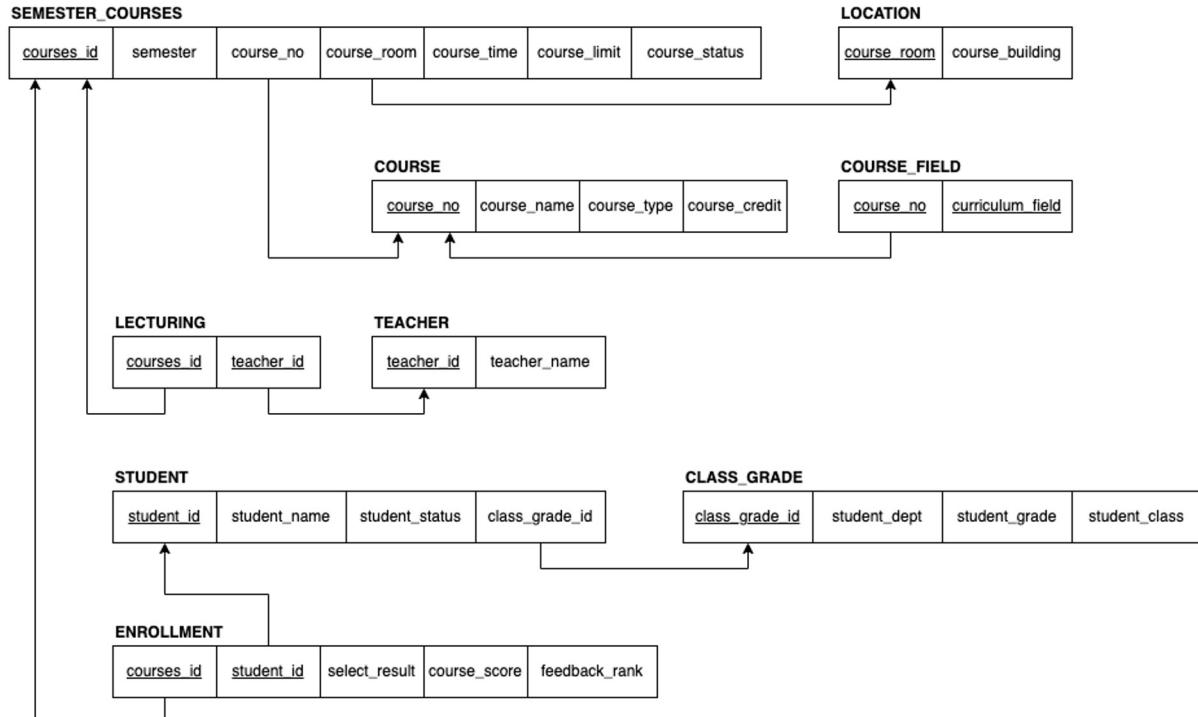
ENROLLMENT

欄位名稱	資料型態	相依資訊	欄位說明			
courses_id	varchar(10)	主鍵；外來鍵	當學期課程編號	C0001	C0002	C0003
student_id	varchar(10)	主鍵；外來鍵	學生編號	S0001	S0002	S0003
select_result	varchar(10)		選課結果	中選	中選	落選
course_score	NUMERIC		學期總成績	90	63	-
feedback_rank	INTEGER		教學評量結果 (1-5分)	5	2	-

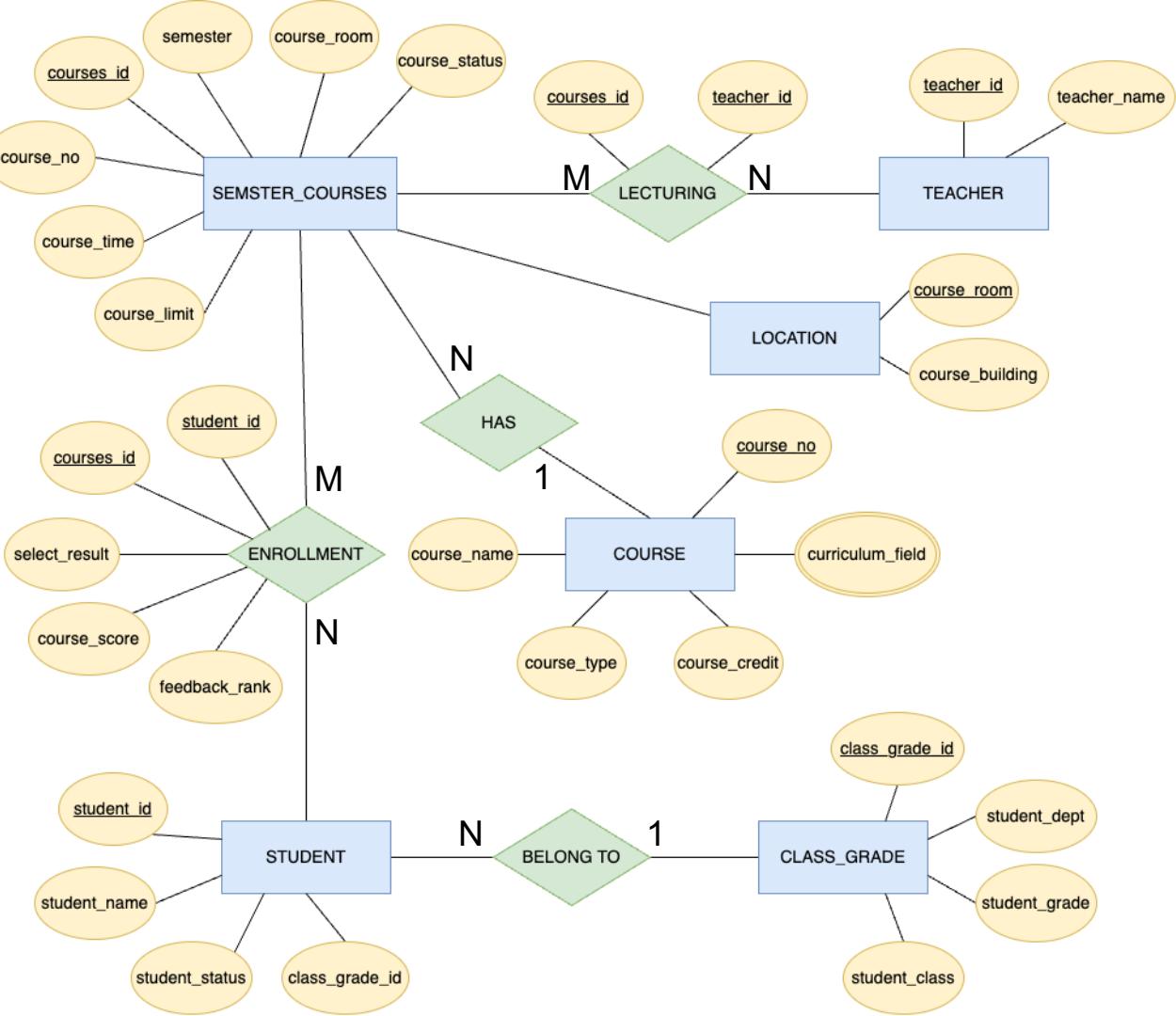
第二題

ER-Diagram

ER-Diagram



ER-Diagram



第三題

SQL解決下列問題

第一小題

1. 「A0001微積分」上課地點要由K205修改到K210 大教室，課程人數限制增加至200人，該怎麼做？

```
第三題第1小題.sql M ×  
第三題 > 第三題第1小題.sql  
1 | INSERT INTO LOCATION (course_room, course_building)  
2 | VALUES ("K210", "工程一館");  
3 |  
4 | UPDATE SEMESTER_COURSES  
5 | SET course_room = "K210", course_limit = 200  
6 | WHERE course_no = "A0001";|  
AND semester = "1112";
```

第二小題

2. 請設計一個方便助教點名的點名表，欄位自行定義，產出結果以「A0002 計算機概論」課程為範例。

```
1 | CREATE TABLE IF NOT EXISTS ROLL_CALL_LIST AS
2 | SELECT
3 | course_no||" "||course_name AS "課程",
4 | student_name AS "學生",
5 | student_dept||student_grade||student_class AS "系所"
6 | FROM SEMESTER_COURSES
7 | NATURAL JOIN COURSE
8 | NATURAL JOIN ENROLLMENT
9 | NATURAL JOIN STUDENT
10 | NATURAL JOIN CLASS_GRADE
11 | WHERE course_no = "A0002"
12 | AND student_status = "在學"
13 | AND select_result <> "落選"
14 |
15 | ALTER TABLE ROLL_CALL_LIST
16 | ADD "Week 1" varchar(1);
17 | ALTER TABLE ROLL_CALL_LIST
18 | ADD "Week 2" varchar(1);
19 | ALTER TABLE ROLL_CALL_LIST
20 | ADD "Week 3" varchar(1);
21 | ALTER TABLE ROLL_CALL_LIST
22 | ADD "Week 4" varchar(1);
23 | ALTER TABLE ROLL_CALL_LIST
24 | ADD "Week 5" varchar(1);
25 | ALTER TABLE ROLL_CALL_LIST
26 | ADD "Week 6" varchar(1);
27 | ALTER TABLE ROLL_CALL_LIST
```

Database Structure Browse Data Edit Pragmas 執行 SQL

Table: ROLL_CALL_LIST Filter in any column

課程	學生	系所	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	W
通過	通過	通過	通過	通過	通過	通過	通過	通過	通過	通過	通過	通過	通過	過
1 A0002 計算機概論	周瑜	數學系1A	NULL	NULL	NL									
2 A0002 計算機概論	黃益	數學系1A	NULL	NULL	NL									
3 A0002 計算機概論	趙宴	數學系1A	NULL	NULL	NL									
4 A0002 計算機概論	夏侯惇	數學系1A	NULL	NULL	NL									
5 A0002 計算機概論	關羽	資訊工程系1A	NULL	NULL	NL									
6 A0002 計算機概論	草蛇	資訊工程研究所1A	NULL	NULL	NL									

第三小題

3. 課程成績不及格的學生比例資料 (大學部 : 低於60分、碩博 : 70分)

```
1 CREATE VIEW "VIEW_不及格人次" AS
2 SELECT courses_id, COUNT(*) AS "不及格人次"
3 FROM ENROLLMENT
4 NATURAL JOIN STUDENT
5 NATURAL JOIN CLASS_GRADE
6 WHERE (
7   | (student_dept LIKE '%研究所' OR student_dept LIKE '%碩士班') AND course_score < 70
8 )
9 OR (
10  | (student_dept NOT LIKE '%研究所' AND student_dept NOT LIKE '%碩士班') AND
11    course_score < 60
12 )
13 GROUP BY courses_id
14 ;
```

Database Structure Browse Data

Table:

courses_id	不及格人次
1 C0001	3
2 C0002	2
3 C0003	3
4 C0004	2
5 C0005	1
6 C0006	3
7 C0007	2

第三小題

3. 課程成績不及格的學生比例資料 (大學部 : 低於60分、碩博 : 70分)

```
14  
15 CREATE VIEW "VIEW_修課人次" AS  
16 SELECT courses_id, COUNT(*) AS "修課人次"  
17 FROM ENROLLMENT  
18 WHERE course_score IS NOT NULL  
19 GROUP BY courses_id  
20 ;  
21
```

Table: VIEW_修課人次

courses_id	修課人次	
	過濾	過濾
1 C0001		6
2 C0002		6
3 C0003		9
4 C0004		5
5 C0005		6
6 C0006		9
7 C0007		5

第三小題

3. 課程成績不及格的學生比例資料 (大學部 : 低於60分、碩博 : 70分)

```
20  
21 SELECT  
22 course_name AS "課名",  
23 teacher_name AS "授課教師",  
24 "不及格人次",  
25 "修課人次",  
26 ROUND(CAST("不及格人次" AS float)/CAST("修課人次" AS float)*100, 2) AS "不及格比例(%)"  
27 FROM SEMESTER_COURSES  
28 NATURAL JOIN COURSE  
29 NATURAL JOIN LECTURING  
30 NATURAL JOIN TEACHER  
31 NATURAL JOIN "VIEW_不及格人次"  
32 NATURAL JOIN "VIEW_修課人次"  
33 ;
```

	課名	授課教師	不及格人次	修課人次	不及格比例(%)
1	微積分	岳飛	3	6	50.0
2	計算機概論	陸羽	2	6	33.33
3	統計學習	劉邦	3	9	33.33
4	統計學習	項羽	3	9	33.33
5	經濟學	孔丘	2	5	40.0
6	統計學	莊周	1	6	16.67
7	音樂欣賞	巴哈	3	9	33.33
8	演算法	達文西	2	5	40.0

第四小題

4. 列出一張大表，供教務處了解各系學生修課領域分佈情況

```
1 CREATE VIEW "VIEW_DEPT_FIELD_COUNT" AS
2 SELECT
3 student_dept||student_grade||student_class AS "學生系所",
4 curriculum_field AS "課程領域",
5 COUNT(*) AS "人次"
6 FROM SEMESTER_COURSES
7 NATURAL JOIN COURSE
8 NATURAL JOIN COURSE_FIELD
9 NATURAL JOIN ENROLLMENT
10 NATURAL JOIN STUDENT
11 NATURAL JOIN CLASS_GRADE
12 WHERE select_result <> "落選"
13 AND course_status <> "停開"
14 GROUP BY "學生系所", "課程領域"
15 ORDER BY "學生系所"
16 ;
17
```

學生系所	課程領域	人文
數學系IA	人工智慧	4
數學系IA	人文思想	4
數學系IA	基礎知識	12
數學系IA	理論數學	7
數學系IA	統計推論	3
數學系IA	財務工程	3
數學系碩士班IA	人工智慧	2
數學系碩士班IA	統計推論	2
數學系碩士班IA	財務工程	2
數學系碩士班IA	資料科學	2
資訊工程研究所IA	人工智慧	6
資訊工程研究所IA	人文思想	4
資訊工程研究所IA	基礎知識	3
資訊工程研究所IA	統計推論	5
資訊工程研究所IA	財務工程	5
資訊工程研究所IA	資料科學	4
資訊工程系IA	人工智慧	3
資訊工程系IA	人文思想	2
資訊工程系IA	基礎知識	5
資訊工程系IA	理論數學	1
資訊工程系IA	統計推論	1
資訊工程系IA	財務工程	1
資訊工程系IA	資料科學	1
資訊管理系IA	人文思想	1
資訊管理系IA	基礎知識	1

第四小題

```
15  
16 CREATE VIEW "VIEW_DEPT_FIELD_SUM" AS  
17 SELECT "學生系所", SUM("人次") AS "總人次"  
18 FROM VIEW_DEPT_FIELD_COUNT  
19 GROUP BY "學生系所"  
20 ;  
  
21  
22 SELECT  
23 "學生系所",  
24 "課程領域",  
25 "人次",  
26 ROUND(CAST("人次" AS float)/CAST("總人次" AS float)*100, 2) AS "佔比(%)"  
27 FROM VIEW_DEPT_FIELD_COUNT  
28 NATURAL JOIN VIEW_DEPT_FIELD_SUM  
29 ORDER BY "學生系所", "佔比(%)" DESC  
30 ;
```

Database Structure Browse Data Edit

Table: VIEW_DEPT_FIELD_SUM

學生系所	總人次
過濾	過濾
1 數學系1A	33
2 數學系碩士班1A	8
3 資訊工程研究所1A	27
4 資訊工程系1A	14
5 資訊管理系1A	2

學生系所	課程領域	人次	佔比(%)
數學系1A	基礎知識	12	36.36
數學系1A	理論數學	7	21.21
數學系1A	人工智慧	4	12.12
數學系1A	人文思想	4	12.12
數學系1A	統計推論	3	9.09
數學系1A	財務工程	3	9.09
數學系碩士班1A	人工智慧	2	25.0
數學系碩士班1A	統計推論	2	25.0
數學系碩士班1A	財務工程	2	25.0
數學系碩士班1A	資料科學	2	25.0
資訊工程研究所1A	人工智慧	6	22.22
資訊工程研究所1A	統計推論	5	18.52
資訊工程研究所1A	財務工程	5	18.52
資訊工程研究所1A	人文思想	4	14.81
資訊工程研究所1A	資料科學	4	14.81
資訊工程研究所1A	基礎知識	3	11.11
資訊工程系1A	基礎知識	5	35.71
資訊工程系1A	人工智慧	3	21.43
資訊工程系1A	人文思想	2	14.29
資訊工程系1A	理論數學	1	7.14
資訊工程系1A	統計推論	1	7.14
資訊工程系1A	財務工程	1	7.14
資訊工程系1A	資料科學	1	7.14
資訊管理系1A	人文思想	1	50.0
資訊管理系1A	基礎知識	1	50.0

第五小題

5. 列出教學評量平均分數及總分，依平均分數由高至低排序

第三題 > 第三題第5小題.sql

```
1 SELECT
2 course_no AS "課程編號",
3 course_name AS "課程名稱",
4 teacher_name AS "授課教師",
5 SUM(feedback_rank) AS "評量總分",
6 ROUND(CAST(SUM(feedback_rank) AS float)/COUNT(*), 2) AS "評量平均分數"
7 FROM SEMESTER_COURSES
8 NATURAL JOIN COURSE
9 NATURAL JOIN LECTURING
10 NATURAL JOIN TEACHER
11 NATURAL JOIN ENROLLMENT
12 WHERE feedback_rank IS NOT NULL
13 GROUP BY "課程編號", "授課教師"
14 ORDER BY "評量平均分數" DESC
15 ; |
```

	課程編號	課程名稱	授課教師	評量總分	評量平均分數
1	A0006	音樂欣賞	巴哈	38	4.22
2	A0002	計算機概論	陸羽	24	4.0
3	A0004	經濟學	孔丘	19	3.8
4	A0003	統計學習	劉邦	34	3.78
5	A0003	統計學習	項羽	34	3.78
6	A0007	演算法	達文西	16	3.2
7	A0001	微積分	岳飛	17	2.83
8	A0005	統計學	莊周	17	2.83

第四題

資料庫實作資料庫正規化

實作

```
1 CREATE TABLE IF NOT EXISTS LOCATION (
2 course_room varchar(20),
3 course_building varchar(20),
4 PRIMARY KEY (course_room)
5 );
6
7 CREATE TABLE IF NOT EXISTS COURSE (
8 course_no varchar(10),
9 course_name varchar(255),
10 course_type varchar(10),
11 course_credit INTEGER,
12 PRIMARY KEY (course_no)
13 );
14
15 CREATE TABLE IF NOT EXISTS COURSE_FIELD (
16 course_no varchar(10),
17 curriculum_field TEXT,
18 PRIMARY KEY (course_no, curriculum_field),
19 FOREIGN KEY (course_no) REFERENCES COURSE(course_no)
20 );
21
22 CREATE TABLE IF NOT EXISTS SEMESTER_COURSES (
23 courses_id varchar(10),
24 semester varchar(4),
25 course_no varchar(10),
26 course_room varchar(20),
27 course_time varchar(20),
28 course_limit INTEGER,
29 course_status varchar(10),
30 PRIMARY KEY (courses_id),
31 FOREIGN KEY (course_no) REFERENCES COURSE(course_no),
32 FOREIGN KEY (course_room) REFERENCES LOCATION(course_room)
33 );
34
35 CREATE TABLE IF NOT EXISTS TEACHER (
36 teacher_id varchar(10),
37 teacher_name varchar(20),
38 PRIMARY KEY (teacher_id)
39 );
40
41 CREATE TABLE IF NOT EXISTS LECTURING (
42 courses_id varchar(10),
43 teacher_id varchar(10),
44 PRIMARY KEY (courses_id, teacher_id),
45 FOREIGN KEY (courses_id) REFERENCES SEMESTER_COURSES(courses_id),
46 FOREIGN KEY (teacher_id) REFERENCES TEACHER(teacher_id)
47 );
48
49 CREATE TABLE IF NOT EXISTS CLASS_GRADE (
50 class_grade_id varchar(10),
51 student_dept varchar(30),
52 student_grade INTEGER,
53 student_class varchar(1),
54 PRIMARY KEY (class_grade_id)
55 );
56
57 CREATE TABLE IF NOT EXISTS STUDENT (
58 student_id varchar(10),
59 student_name varchar(20),
60 student_status varchar(10),
61 class_grade_id varchar(10),
62 PRIMARY KEY (student_id),
63 FOREIGN KEY (class_grade_id) REFERENCES CLASS_GRADE(class_grade_id)
64 );
65
66 CREATE TABLE IF NOT EXISTS ENROLLMENT (
67 courses_id varchar(10),
68 student_id varchar(10),
69 select_result varchar(10),
70 course_score NUMERIC,
71 feedback_rank INTEGER,
72 PRIMARY KEY (courses_id, student_id),
73 FOREIGN KEY (courses_id) REFERENCES SEMESTER_COURSES(courses_id),
74 FOREIGN KEY (student_id) REFERENCES STUDENT(student_id)
75 );
```

實作

DB Browser for SQLite - D:\Repositories\Database\第四題\Database.db

檔案(F) 編輯(E) 查看(V) Tools 幫助(H)

新建資料庫(N) 打開資料庫(O) Write Changes Revert Changes Open Project Save Project Attach Database 閉關資料庫(C)

Database Structure Browse Data Edit Pragmas 執行 SQL

Create Table Create Index Print

名稱	類型	架構
資料表 (11)		
CLASS_GRADE		CREATE TABLE CLASS_GRADE (class_grade_id varchar(10), student_dept varchar(30), student_grade INTEGER, student_class varchar(1), PRIMARY KEY (class_grade_id)) "class_grade_id" varchar(10) "student_dept" varchar(30) "student_grade" INTEGER "student_class" varchar(1)
COURSE		CREATE TABLE COURSE (course_no varchar(10), course_name varchar(255), course_type varchar(10), course_credit INTEGER, PRIMARY KEY (course_no)) "course_no" varchar(10) "course_name" varchar(255) "course_type" varchar(10) "course_credit" INTEGER
COURSE_FIELD		CREATE TABLE COURSE_FIELD (course_no varchar(10), curriculum_field TEXT, PRIMARY KEY (course_no, curriculum_field), FOREIGN KEY (course_no) REFERENCES COURSE(course_no)) "course_no" varchar(10) "curriculum_field" TEXT
ENROLLMENT		CREATE TABLE ENROLLMENT (courses_id varchar(10), student_id varchar(10), select_result varchar(10), course_score NUMERIC, feedback_rank INTEGER, PRIMARY KEY (courses_id, student_id), FOREIGN KEY (courses_id) REFERENCES COURSE(course_no), FOREIGN KEY (student_id) REFERENCES STUDENT(student_id)) "courses_id" varchar(10) "student_id" varchar(10) "select_result" varchar(10) "course_score" NUMERIC "feedback_rank" INTEGER
LECTURING		CREATE TABLE LECTURING (courses_id varchar(10), teacher_id varchar(10), PRIMARY KEY (courses_id, teacher_id), FOREIGN KEY (courses_id) REFERENCES SEMESTER_COURSES(courses_id), FOREIGN KEY (teacher_id) REFERENCES TEACHER(teacher_id)) "courses_id" varchar(10) "teacher_id" varchar(10)
LOCATION		CREATE TABLE LOCATION (course_room varchar(20), course_building varchar(20), PRIMARY KEY (course_room)) "course_room" varchar(20) "course_building" varchar(20)
SEMESTER_COURSES		CREATE TABLE SEMESTER_COURSES (courses_id varchar(10), semester varchar(4), course_no varchar(10), course_room varchar(20), course_time varchar(20), course_limit INTEGER, course_status varchar(10), I "courses_id" varchar(10) "semester" varchar(4) "course_no" varchar(10) "course_room" varchar(20) "course_time" varchar(20) "course_limit" INTEGER "course_status" varchar(10)
STUDENT		CREATE TABLE STUDENT (student_id varchar(10), student_name varchar(20), student_status varchar(10), class_grade_id varchar(10), PRIMARY KEY (student_id), FOREIGN KEY (class_grade_id) REFERENCES CLASS_GRADE(class_grade_id)) "student_id" varchar(10) "student_name" varchar(20) "student_status" varchar(10) "class_grade_id" varchar(10)
TEACHER		CREATE TABLE TEACHER (teacher_id varchar(10), teacher_name varchar(20), PRIMARY KEY (teacher_id)) "teacher_id" varchar(10)

實作

```
31 # 接著進行第一正規化(把多值屬性的列拆成多個列)
32 conn = sqlite3.connect(db_name)
33 df_course_data = pd.read_sql("SELECT * FROM course_data", conn)
34
35 columns_with_multivalues = ["teacher_name", "curriculum_field"]
36
37 def first_normalization(df):
38     for column in columns_with_multivalues:
39         for i in range(len(df)):
40
41             # 把欄位的值取出來
42             value = df.loc[i, column]
43
44             # 如果字串裡面包含 ',', 代表第 i 列的這個屬性為多值屬性
45             if(',') in value:
46                 values = value.split(',')
47
48                 # 先將原本第 i 列的這個屬性改成第一個值
49                 df.loc[i, column] = values[0]
50
51                 for j in range(1, len(values)):
52                     # 對針第二個以上的值
53                     # 先建立原本第 i 個列的副本
54                     # 接著把該屬性設為第 j 個值
55                     # 把新的列插入到資料表中
56                     df_temp = pd.DataFrame(data=df, index=[i])
57                     df_temp.loc[i, column] = values[j]
58                     df = pd.concat([df, df_temp], ignore_index=True)
59
60             return df
61 df_first_normalized = first_normalization(df_course_data)
```

實作

```
103 # LOCATION
104 ▼ sql = """
105 INSERT INTO LOCATION (course_room, course_building)
106 SELECT DISTINCT course_room, course_building
107 FROM ''' + table_name_1nf + '''
108 cur.execute(sql)
109
110
111 # COURSE
112 ▼ sql = """
113 INSERT INTO COURSE (course_no, course_name, course_type, course_credit)
114 SELECT DISTINCT course_no, course_name, course_type, course_credit
115 FROM ''' + table_name_1nf + '''
116 cur.execute(sql)
117
118
119 # COURSE_FIELD
120 ▼ sql = """
121 INSERT INTO COURSE_FIELD (course_no, curriculum_field)
122 SELECT DISTINCT course_no, curriculum_field
123 FROM ''' + table_name_1nf + '''
124 cur.execute(sql)
125
126 conn.commit()
127 conn.close()
```

實作

Table: LOCATION

	course_room	course_building
	過濾	過濾
1	K205	工程一館
2	L102	工程五館
3	M-605	鴻經館
4	I1-018	管理二館
5	I1-304	管理二館
6	O-214	綜教館

Database Structure Browse Data Edit Pragmas 執行 SQL

Table: COURSE

	course_no	course_name	course_type	course_credit
	過濾	過濾	過濾	過濾
1	A0001	微積分	必修	2
2	A0002	計算機概論	必修	3
3	A0003	統計學習	選修	3
4	A0004	經濟學	必修	3
5	A0005	統計學	選修	3
6	A0006	音樂欣賞	選修	2
7	A0007	演算法	選修	3

Database Structure Browse Data

Table: COURSE_FIELD

	course_no	curriculum_field
	過濾	過濾
1	A0001	理論數學
2	A0002	基礎知識
3	A0003	財務工程
4	A0004	基礎知識
5	A0005	基礎知識
6	A0006	人文思想
7	A0007	人工智慧
8	A0002	人工智慧
9	A0003	統計推論
10	A0007	資料科學

實作

```
134 # SEMESTER_COURSES
135 conn = sqlite3.connect("Database.db")
136 df_course_data = pd.read_sql("SELECT * FROM " + table_name_1nf, conn)
137 df_semester_courses = df_course_data[["semester", "course_no", "course_room", "course_time", "course_limit", "course_status"]]
138 df_semester_courses = df_semester_courses.drop_duplicates()
139
140 # 新增一個 courses_id 欄位
141 courses_id = []
142 for i in range((len(df_semester_courses))):
143     courses_id.append("C" + '{:04d}'.format(i+1))
144 df_semester_courses.insert(0, "courses_id", courses_id)
145
146 # 將資料表存入 database 中
147 column_dtype = {"courses_id": "varchar(10)", "semester": "varchar(4)", "course_no": "varchar(10)", "course_room": "varchar(20)",
148 "course_time": "varchar(20)", "course_limit": "INTEGER", "course_status": "varchar(10)"}
149 df_semester_courses.to_sql("SEMESTER_COURSES", conn, if_exists="append", index=False, dtype=column_dtype)
150 conn.close()
151
152 # TEACHER
153 conn = sqlite3.connect("Database.db")
154 df_course_data = pd.read_sql("SELECT * FROM " + table_name_1nf, conn)
155 df_teacher = df_course_data[["teacher_name"]]
156 df_teacher = df_teacher.drop_duplicates()
157
158 teacher_id = []
159 for i in range((len(df_teacher))):
160     teacher_id.append("T" + '{:04d}'.format(i+1))
161 df_teacher.insert(0, "teacher_id", teacher_id)
162
163 column_dtype = {"teacher_id": "varchar(10)", "teacher_name": "varchar(20)"}
164 df_teacher.to_sql("TEACHER", conn, if_exists="append", index=False, dtype=column_dtype)
165 conn.close()
```

實作

Database Structure Browse Data Edit Pragmas 執行 SQL

Table: SEMESTER_COURSES

	courses_id	semester	course_no	course_room	course_time	course_limit	course_status
1	C0001	1112	A0001	K205	一567	50	開課
2	C0002	1112	A0002	L102	二34,五4	50	開課
3	C0003	1112	A0003	M-605	四567	50	開課
4	C0004	1112	A0004	I1-018	四567	50	開課
5	C0005	1112	A0005	I1-304	五234	50	開課
6	C0006	1112	A0006	O-214	三56	100	開課
7	C0007	1112	A0007	L102	三234	50	開課

Database Structure Browse Data

Table: TEACHER

	teacher_id	teacher_name
1	T0001	岳飛
2	T0002	陸羽
3	T0003	劉邦
4	T0004	孔丘
5	T0005	莊周
6	T0006	巴哈
7	T0007	達文西
8	T0008	項羽

實作

```
166 # LECTURING
167 conn = sqlite3.connect("Database.db")
168 df_course_data = pd.read_sql("SELECT * FROM " + table_name_1nf, conn)
169 df_semester_courses = pd.read_sql("SELECT * FROM SEMESTER_COURSES", conn)
170 df_teacher = pd.read_sql("SELECT * FROM TEACHER", conn)
171
172 # 先在 1nf 後的大表中分別新增一個 courses_id 以及一個 teacher_id 欄位
173 # (但僅是在 dataframe 上操作，新增的欄位並無 commit 到 database 裡)
174 courses_id = []
175 for i in range(len(df_course_data)):
176     # 先取出大表中第 i 列的 semester 以及 course_no 欄位
177     semester = df_course_data.loc[i, "semester"]
178     course_no = df_course_data.loc[i, "course_no"]
179     # 接著從 SEMESTER_COURSES 中，找到 (semester, course_no) 所對應的 courses_id
180     id = df_semester_courses.loc[(df_semester_courses["semester"] == semester) & (df_semester_courses["course_no"] == course_no), "courses_id"]
181     id = id.values[0]
182     # 把 id 新增到 list 中
183     courses_id.append(id)
184 # 在大表的 dataframe 新增一個 courses_id 欄位
185 df_course_data.insert(0, "courses_id", courses_id)
186
187 teacher_id = []
188 for i in range(len(df_course_data)):
189     teacher_name = df_course_data.loc[i, "teacher_name"]
190     id = df_teacher.loc[df_teacher["teacher_name"] == teacher_name, "teacher_id"]
191     id = id.values[0]
192     teacher_id.append(id)
193 df_course_data.insert(0, "teacher_id", teacher_id)
194
195 # 接著從大表的 dataframe 中取出 LECTURING 所需的欄位
196 # 並寫入到 database 中的 table
197 df_lecturing = df_course_data[["courses_id", "teacher_id"]]
198 df_lecturing = df_lecturing.drop_duplicates()
199 column_dtype = {"courses_id": "varchar(10)", "teacher_id": "varchar(10)"}
200 df_lecturing.to_sql("LECTURING", conn, if_exists="append", index=False, dtype=column_dtype)
201 conn.close()
```

Database Structure Browse Data

Table: LECTURING

	courses_id	teacher_id
	過濾	過濾
1	C0001	T0001
2	C0002	T0002
3	C0003	T0003
4	C0004	T0004
5	C0005	T0005
6	C0006	T0006
7	C0007	T0007
8	C0003	T0008

實作

```
203 # CLASS_GRADE
204 conn = sqlite3.connect("Database.db")
205 df_course_data = pd.read_sql("SELECT * FROM " + table_name_1nf, conn)
206 df_class_grade = df_course_data[["student_dept", "student_grade", "student_class"]]
207 df_class_grade = df_class_grade.drop_duplicates()
208
209 class_grade_id = []
210 for i in range((len(df_class_grade))):
211     class_grade_id.append("CG" + '{:04d}'.format(i+1))
212 df_class_grade.insert(0, "class_grade_id", class_grade_id)
213
214 column_dtype = {"class_grade_id": "varchar(10)", "student_dept": "varchar(30)", "student_grade": "INTEGER", "student_class": "varchar(1)"}
215 df_class_grade.to_sql("CLASS_GRADE", conn, if_exists="append", index=False, dtype=column_dtype)
216 conn.close()
```

Database Structure Browse Data Edit Pragmas 執行 SQL

Table: CLASS_GRADE

class_grade_id	student_dept	student_grade	student_class
過濾	過濾	過濾	過濾
1 CG0001	數學系		1 A
2 CG0002	資訊工程系		1 A
3 CG0003	資訊工程研究所		1 A
4 CG0004	資訊管理系		1 A
5 CG0005	數學系碩士班		1 A

實作

```
218 # STUDENT
219 conn = sqlite3.connect("Database.db")
220 df_course_data = pd.read_sql("SELECT * FROM " + table_name_1nf, conn)
221 df_class_grade = pd.read_sql("SELECT * FROM CLASS_GRADE", conn)
222
223 class_grade_id = []
224 for i in range((len(df_course_data))):
225     student_dept = df_course_data.loc[i, "student_dept"]
226     student_grade = df_course_data.loc[i, "student_grade"]
227     student_class = df_course_data.loc[i, "student_class"]
228     id = df_class_grade.loc[(df_class_grade["student_dept"] == student_dept) & (df_class_grade["student_grade"] == student_grade) &
229     (df_class_grade["student_class"] == student_class), "class_grade_id"]
230     id = id.values[0]
231     class_grade_id.append(id)
232     df_course_data.insert(0, "class_grade_id", class_grade_id)
233
234 df_student = df_course_data[["student_name", "student_status", "class_grade_id"]]
235 df_student = df_student.drop_duplicates()
236
237 student_id = []
238 for i in range((len(df_student))):
239     student_id.append("S" + '{:04d}'.format(i+1))
240 df_student.insert(0, "student_id", student_id)
241 column_dtype = {"student_id": "varchar(10)", "student_name": "varchar(20)", "student_status": "varchar(10)", "class_grade_id": "varchar(10)"}
242 df_student.to_sql("STUDENT", conn, if_exists="append", index=False, dtype=column_dtype)
243 conn.close()
```

實作

Database Structure Browse Data Edit Pragmas 執行 SQL

Table: STUDENT

過濾

	student_id	student_name	student_status	class_grade_id
1	S0001	張飛	在學	CG0001
2	S0002	孫尚香	休學	CG0001
3	S0003	周瑜	在學	CG0001
4	S0004	黃蓋	在學	CG0001
5	S0005	趙雲	在學	CG0001
6	S0006	關興	在學	CG0001
7	S0007	夏侯惇	在學	CG0001
8	S0008	龐統	休學	CG0002
9	S0009	關羽	在學	CG0002
10	S0010	華雄	退學	CG0003
11	S0011	華陀	在學	CG0003
12	S0012	劉備	在學	CG0004
13	S0013	呂布	在學	CG0003
14	S0014	諸葛亮	在學	CG0003
15	S0015	呂蒙	在學	CG0003
16	S0016	圖靈	在學	CG0005
17	S0017	巴斯卡	在學	CG0005
18	S0018	大喬	在學	CG0002
19	S0019	甘寧	在學	CG0002
20	S0020	司馬昭	在學	CG0002
21	S0021	馬超	在學	CG0002
22	S0022	郭嘉	在學	CG0003

實作

```
245 # ENROLLMENT
246 conn = sqlite3.connect("Database.db")
247 df_course_data = pd.read_sql("SELECT * FROM " + table_name_1nf, conn)
248 df_semester_courses = pd.read_sql("SELECT * FROM SEMESTER_COURSES", conn)
249 df_class_grade = pd.read_sql("SELECT * FROM CLASS_GRADE", conn)
250 df_student = pd.read_sql("SELECT * FROM STUDENT", conn)
251
252 courses_id = []
253 for i in range((len(df_course_data))):
254     semester = df_course_data.loc[i, "semester"]
255     course_no = df_course_data.loc[i, "course_no"]
256     id = df_semester_courses.loc[(df_semester_courses["semester"] == semester) & (df_semester_courses["course_no"] == course_no), "courses_id"]
257     id = id.values[0]
258     courses_id.append(id)
259 df_course_data.insert(0, "courses_id", courses_id)
260
261 class_grade_id = []
262 for i in range((len(df_course_data))):
263     student_dept = df_course_data.loc[i, "student_dept"]
264     student_grade = df_course_data.loc[i, "student_grade"]
265     student_class = df_course_data.loc[i, "student_class"]
266     id = df_class_grade.loc[(df_class_grade["student_dept"] == student_dept) & (df_class_grade["student_grade"] == student_grade) & (df_class_grade["student_class"] == student_class), "class_grade_id"]
267     id = id.values[0]
268     class_grade_id.append(id)
269 df_course_data.insert(0, "class_grade_id", class_grade_id)
270
271 student_id = []
272 for i in range((len(df_course_data))):
273     student_name = df_course_data.loc[i, "student_name"]
274     student_status = df_course_data.loc[i, "student_status"]
275     class_grade_id = df_course_data.loc[i, "class_grade_id"]
276     id = df_student.loc[(df_student["student_name"] == student_name) & (df_student["student_status"] == student_status) & (df_student["class_grade_id"] == class_grade_id), "student_id"]
277     id = id.values[0]
278     student_id.append(id)
279 df_course_data.insert(0, "student_id", student_id)
280
281 df_enrollment = df_course_data[["courses_id", "student_id", "select_result", "course_score", "feedback_rank"]]
282 df_enrollment = df_enrollment.drop_duplicates()
283 column_dtype = {"courses_id": "varchar(10)", "student_id": "varchar(10)", "select_result": "varchar(10)", "course_score": "NUMERIC", "feedback_rank": "INTEGER"}
284 df_enrollment.to_sql("ENROLLMENT", conn, if_exists="append", index=False, dtype=column_dtype)
285 conn.close()
```

實作

courses_id	student_id	select_result	course_score	feedback_rank
C0001	S0001	中選	77.7	1
C0001	S0002	中選		
C0001	S0003	中選	56	2
C0001	S0004	中選	34	3
C0001	S0005	中選	98	4
C0001	S0006	中選	55	5
C0001	S0007	中選	67	2
C0001	S0008	中選		
C0002	S0009	中選	66	3
C0002	S0008	中選		
C0002	S0003	中選	93	4
C0002	S0004	中選	44	4
C0002	S0005	中選	49	5
C0002	S0007	中選	78	3
C0002	S0010	中選		
C0002	S0011	中選	74	5
C0003	S0008	中選		
C0003	S0001	中選	46	5
C0003	S0004	中選	76	4
C0003	S0006	中選	87	5
C0003	S0012	落選		
C0003	S0010	中選		
C0003	S0013	中選	76	4
C0003	S0011	中選	80	5
C0003	S0014	中選	78	3
C0003	S0015	中選	65	4
C0003	S0016	中選	99	3
C0004	S0017	人工加選	69	1
C0004	S0008	落選		
C0004	S0001	中選	56	3
C0004	S0002	中選		
C0004	S0004	中選	67.5	5
C0004	S0005	中選	78	4
C0004	S0007	中選	89	2
C0004	S0013	中選	45	5
C0005	S0009	中選	68.7	1
C0005	S0018	中選	63	4
C0005	S0019	中選	46	5
C0005	S0002	中選		
C0005	S0003	中選	78	2
C0005	S0004	中選	87	3
C0005	S0012	中選	96	2



THE END

THANKS