

“正規化是存放資料的第一步!”

## 資料庫正規形式

甚麼是資料庫的正規化?

為甚麼要正規化?

有幾種形式的正規化?第一、第二、第三正規化

要如何做正規化?

- Foreign Key 外來鍵 (外部鍵)

這個欄位會存放其他資料表的主鍵，主要用來確定資料的參考完整性，只有經過確認的資料才能輸入，避免資料在建立時，因為其他資料不完整而導致資料完整性有缺陷。外來鍵的資料來源也可以是自己本身的主鍵，例如員工資料表裡面的主管編號，這就是一個外來鍵，裡面的資料就是參考本身的員工編號。當輸入員工主管的編號時，會去尋找該主鍵是否存在，確保資料完整性。

## Lab1:學生-導師-修課資料庫範例

未正規化的資料表：

學號	導師	導師辦公室	課程 1	課程 2	課程 3
1022	Jones	412	101-07	143-01	159-02
4123	Smith	216	201-01	211-02	214-01

### 1. 第一正規形式：沒有重複的群組

學號	導師	導師辦公室	課程 #
1022	Jones	412	101-07
1022	Jones	412	143-01
1022	Jones	412	159-02
4123	Smith	216	201-01
4123	Smith	216	211-02
4123	Smith	216	214-01

## 2. 第二正規形式：刪除重複的資料

請注意上述資料表中，每個 [學號] 值都會配對多個 [課程 #] 值。由於 [課程 #] 在運用時並不會依賴 [學號] (主索引鍵)，因此這個關係並非第二正規形式。

學生：

學號	導師	導師辦公室
1022	Jones	412
4123	Smith	216

註冊課程：

學號	課程 #
1022	101-07
1022	143-01
1022	159-02
4123	201-01
4123	211-02
4123	214-01

### 3. 第三正規形式：刪除不依賴索引鍵的資料

[導師辦公室]在運用時會依賴 [導師] 屬性。而解決的方法，就是將該屬性從 [學生] 資料表移至 [教職員] 資料表，如下所示：

學生：

學號	導師
1022	Jones
4123	Smith

教職員：

名稱	辦公室	部門
Jones	412	42
Smith	216	42

來源資料:<https://support.microsoft.com/zhtw/kb/283878/zhtw>

## Lab2:學生-導師-修課資料庫新增實作

注意:需要按照順序新增資料:先有學生、老師資料，才能有老師-學生資料

1. 請在 SQL Manager Tool 中新增 student table 欄位如下:

#	名稱	數據類型	長度/設置
1	student_id	VARCHAR	10
2	name	VARCHAR	20
3	phone	VARCHAR	10

2. 新增 student table 內容 rows 如下:

student_id	name	phone
u001	王曉明	0935875178
u002	李大同	0933551246
u003	孫小毛	0965521126

3. 新增 Teacher table 欄位如下:

#	名稱	數據類型	長度/設置
1	Teacher_id	VARCHAR	10
2	Name	VARCHAR	50
3	Room	VARCHAR	10
4	Dept	VARCHAR	10

4. 新增 Teacher table 內容 rows 如下:

Teacher_id	Name	Room	Dept
t01	瓊斯	412	42
t02	史密斯	216	42

5. 新增 Student-Teacher table 欄位如下:

#	名稱	數據類型	長度/設置
1	Student_id	VARCHAR	10
2	Teacher_id	VARCHAR	10

定義外部鍵

 基本	 選項	 索引	 外鍵	 Partitions	 CREATE
 添加	字段	關聯表	外聯字段	UPDATE時	
 刪除	 Teacher_id	teacher	Teacher_id	RESTRICT	
 清除	 Student_id	student	student_id	RESTRICT	

student\_id, teacher\_id 同時具有key的身分、外部鍵的身分

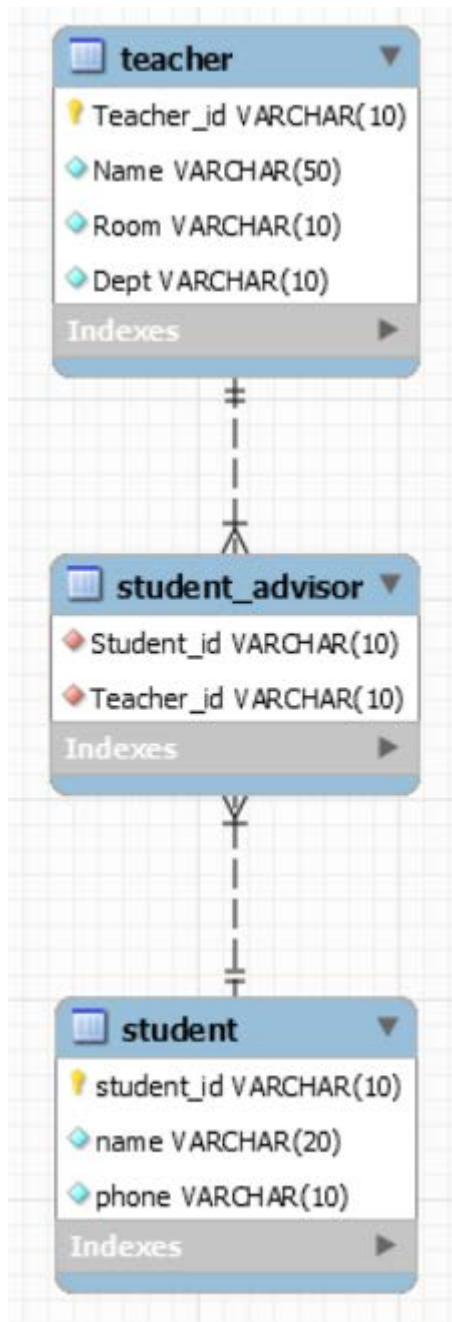
6. 新增 Student-Teacher table 內容 rows 如下:

老師重複出現多筆，表示一位老師有多位學生

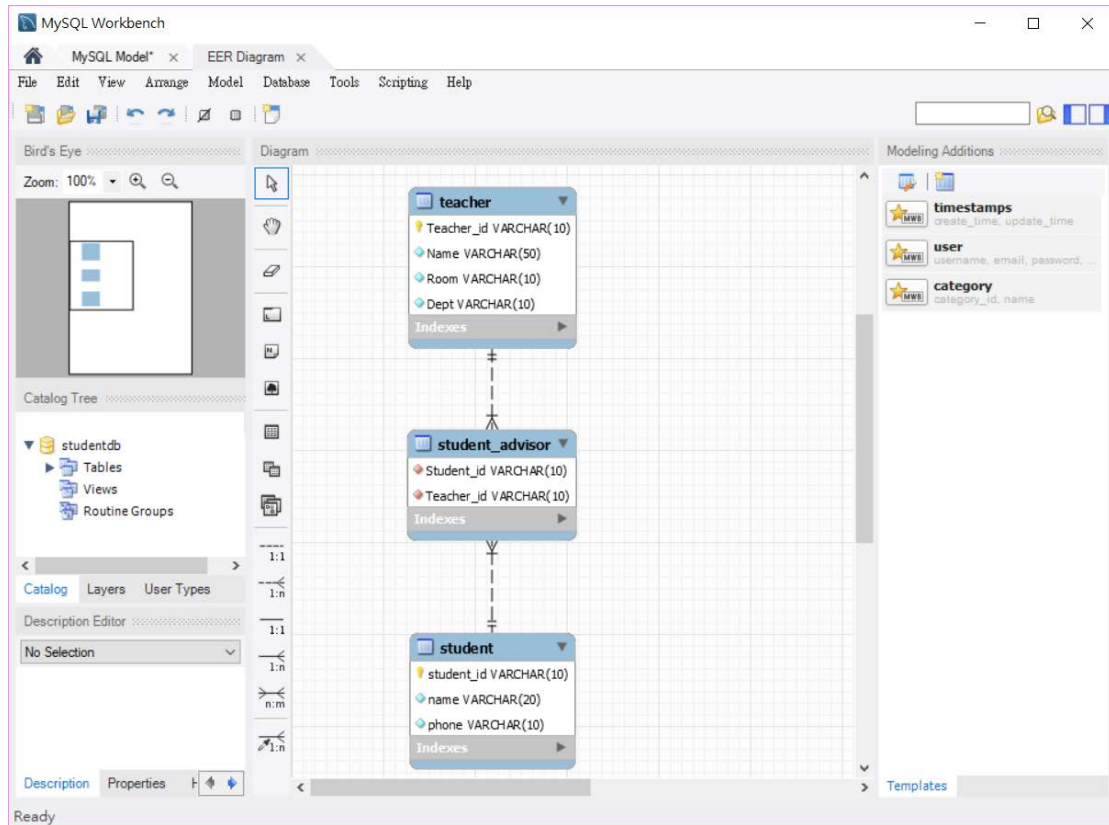
▲ Student_id	▼ Teacher_id
u001	t01
u002	t01
u003	t02

學生也可以重複出現多筆，表示一個學生有兩(多)個導師

▲ Student_id	▼ Teacher_id
u001	t01
u002	t01
u003	t02
u003	t01



ER-Diagram (Entity-Relationship)如何繪製？  
在MySQL workbench中繪製很方便。請[參考影片](#)



請解釋:

多對1

1對1

多對多

這兩張表的關係是:多對1

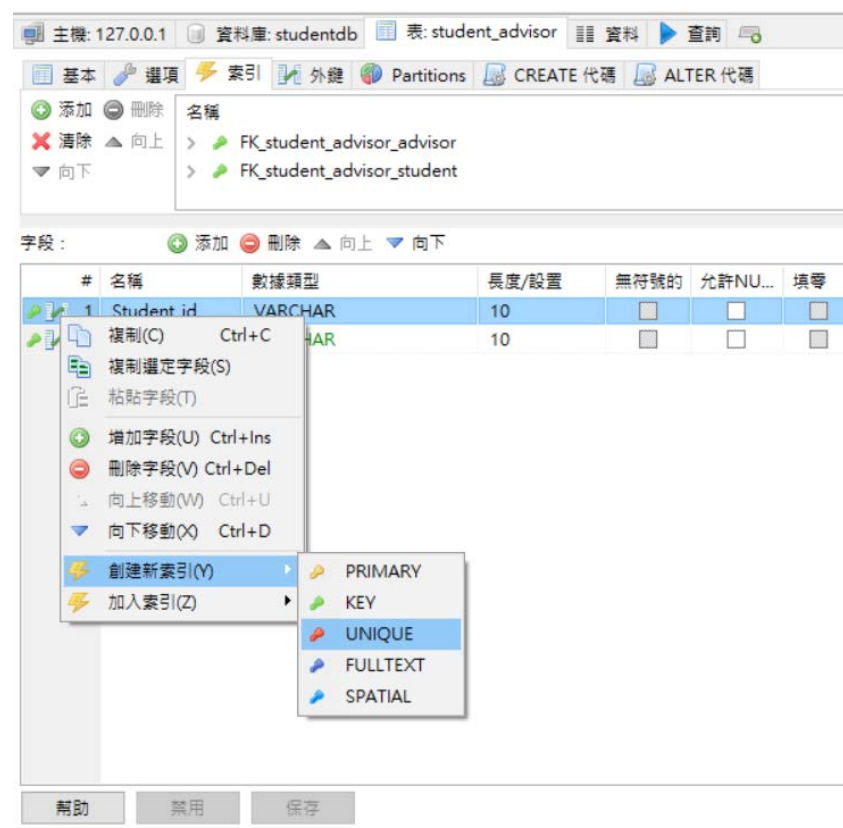
Student_advisor			Student	
Student_id	Teacher_id	student_id	name	phone
u001	t01	u001	王曉明	0935875178
u002	t01	u002	李大同	0933551246
u003	t02	u003	孫小毛	0965521126
u003	t01			



這兩張表的關係是:多對1

Student_advisor		Teacher			
Student_id	Teacher_id	Teacher_id	Name	Room	Dept
u001	t01	t01	瓊斯	412	42
u002	t01	t02	史密斯	216	42
u003	t02				
u003	t01				

請問:如果學校規定一位學生只能有一位導師,也就是學生不可以重複出現多筆,該如何定義鍵值?



增加一個unique唯一的身分

這兩張表的關係是:1對1

Student_advisor		Student		
Student_id	Teacher_id	student_id	name	phone
u001	t01	u001	王曉明	0935875178
u002	t01	u002	李大同	0933551246
u003	t02	u003	孫小毛	0965521126

## Lab3:學生-導師-修課資料庫基本操作

Update, Insert, Delete, Select 的用法:

多個表格 Join 的用法:

查詢 學生—導師 資料

```
select student.student_id as '學號', student.name,
student_advisor.Teacher_id as 'advisor' from student inner join
student_advisor on student.student_id=student_advisor.Student_id;
```

學號	name	advisor
u001	王曉明	t01
u002	李大同	t01
u003	孫小毛	t02

三張表 join 在一起!



學號	姓名	老師
u001	王曉明	瓊斯
u002	李大同	瓊斯
u003	孫小毛	史密斯

```
SELECT student.student_id AS '學號', student.name AS '姓名', teacher.Name AS '老師'
```

```
FROM student_advisor
INNER JOIN student ON student_advisor.Student_id = student.student_id
INNER JOIN teacher ON student_advisor.Teacher_id = teacher.Teacher_id;
```



## Lab4: 增加 class, student\_class 兩個表格

### Class 表格





 class_id	class_name	credit	 teacher_id
<b>101-07</b>	Java程式設計	3	t01
<b>275-08</b>	C#程式設計	3	t02
<b>579-02</b>	Java程式進階	3	t01

### student\_class 學生修課與分數表格

進行匯入資料：不要一行一行的輸入，請執行老師給的 SQL 命令。

 student_id	 class_id	score
u001	101-07	80
u001	275-08	75
u001	579-02	68
u002	101-07	72
u002	275-08	77
u003	101-07	62
u003	579-02	55

### 查詢 學生—修課 資料

 student_id	 class_id	 class_id	class_name	credit	 teacher_id
u001	101-07	<b>101-07</b>	Java程式設計	3	t01
u001	275-08	<b>275-08</b>	C#程式設計	3	t02
u001	579-02	<b>579-02</b>	Java程式進階	3	t01
u002	101-07	<b>101-07</b>	Java程式設計	3	t01
u002	275-08	<b>275-08</b>	C#程式設計	3	t02
u003	101-07	<b>101-07</b>	Java程式設計	3	t01
u003	579-02	<b>579-02</b>	Java程式進階	3	t01

```
SELECT *
FROM student_class
INNER JOIN class ON student_class.class_id = class.class_id
```

student_id	class_name
u001	Java程式設計
u001	C#程式設計
u001	Java程式進階

```
SELECT student_id, class_name
FROM student_class
INNER JOIN class ON student_class.class_id = class.class_id
WHERE student_class.student_id='u001';
```

student_id	class_name	score
u001	Java程式設計	80
u001	C#程式設計	75
u001	Java程式進階	68

```
SELECT student_id, class_name, score
FROM student_class
INNER JOIN class ON student_class.class_id = class.class_id
WHERE student_class.student_id='u001';
```

student_id	name	class_name	score
u001	王曉明	Java程式設計	80
u001	王曉明	C#程式設計	75
u001	王曉明	Java程式進階	68

```
SELECT student_class.student_id, student.name, class_name, score
FROM student_class
Inner join student on student_class.student_id = student.student_id
INNER JOIN class ON student_class.class_id = class.class_id
WHERE student_class.student_id='u001';
```

學生—修課 科目數目 總平均

修課數目	平均	總學分
3	74.3333	9

```
SELECT Count(*) as 修課數目, AVG(score) as 平均, SUM(credit) as 總學分
FROM student_class
INNER JOIN class on student_class.class_id = class.class_id
WHERE student_id='u001'
```

某位導師的所有學生:

	student_id	name
	u001	千曉明1
	u002	李大同

```
SELECT
    student_advisor.student_id, student.name
FROM
    studentdb.student_advisor
    INNER JOIN
    student ON student_advisor.student_id = student.student_id
WHERE
    student_advisor.teacher_id = 't01';
```

## Lab5: 一氣呵成，使用 SQL command

```
--studentdb2 的資料庫結構
CREATE DATABASE IF NOT EXISTS `studentdb2`;
USE `studentdb2`;

-- 表 studentdb.student 結構
```

```
CREATE TABLE IF NOT EXISTS `student` (  
  `student_id` varchar(10) NOT NULL,  
  `name` varchar(20) NOT NULL,  
  `phone` varchar(10) NOT NULL,  
  PRIMARY KEY (`student_id`)  
);  
  
-- 表 studentdb.student 的資料  
INSERT INTO `student` (`student_id`, `name`, `phone`) VALUES  
  ('u001', '王曉明', '0935875178'),  
  ('u002', '李大同', '0933551246'),  
  ('u003', '孫小毛', '0965521126');  
  
-- 表 studentdb.teacher 結構  
CREATE TABLE IF NOT EXISTS `teacher` (  
  `Teacher_id` varchar(10) NOT NULL,  
  `Name` varchar(50) NOT NULL,  
  `Room` varchar(10) NOT NULL,  
  `Dept` varchar(10) NOT NULL,  
  PRIMARY KEY (`Teacher_id`)  
);  
  
-- 表 studentdb.teacher 的資料  
INSERT INTO `teacher` (`Teacher_id`, `Name`, `Room`, `Dept`) VALUES  
  ('t01', '瓊斯', '412', '42'),  
  ('t02', '史密斯', '216', '42');  
  
-- 表 studentdb.class 結構  
CREATE TABLE IF NOT EXISTS `class` (  
  `class_id` varchar(10) NOT NULL,  
  `class_name` varchar(50) NOT NULL,  
  `credit` int(3) unsigned NOT NULL,  
  `teacher_id` varchar(10) NOT NULL,  
  PRIMARY KEY (`class_id`),  
  KEY `FK_class_teacher` (`teacher_id`),  
  CONSTRAINT `FK_class_teacher` FOREIGN KEY (`teacher_id`)  
REFERENCES `teacher` (`Teacher_id`)
```

```
);

-- 表 studentdb.class 的資料
INSERT INTO `class` (`class_id`, `class_name`, `credit`, `teacher_id`) VALUES
    ('101-07', 'Java 程式設計', 3, 't01'),
    ('275-08', 'C#程式設計', 3, 't02'),
    ('579-02', 'Java 程式進階', 3, 't01');

-- 表 studentdb.student_advisor 結構
CREATE TABLE IF NOT EXISTS `student_advisor` (
    `Student_id` varchar(10) NOT NULL,
    `Teacher_id` varchar(10) NOT NULL,
    KEY `FK_student_advisor_advisor` (`Teacher_id`),
    KEY `FK_student_advisor_student` (`Student_id`),
    CONSTRAINT `FK_student_advisor_advisor` FOREIGN KEY (`Teacher_id`)
REFERENCES `teacher` (`Teacher_id`),
    CONSTRAINT `FK_student_advisor_student` FOREIGN KEY (`Student_id`)
REFERENCES `student` (`student_id`)
);

-- 表 studentdb.student_advisor 的資料
INSERT INTO `student_advisor` (`Student_id`, `Teacher_id`) VALUES
    ('u001', 't01'),
    ('u002', 't01'),
    ('u003', 't02');

-- 表 studentdb.student_class 結構
CREATE TABLE IF NOT EXISTS `student_class` (
    `student_id` varchar(10) NOT NULL,
    `class_id` varchar(10) NOT NULL,
    `score` int(3) unsigned NOT NULL,
    KEY `class_id` (`class_id`),
    KEY `FK_student_class_student` (`student_id`),
    CONSTRAINT `FK_student_class_class` FOREIGN KEY (`class_id`)
REFERENCES `class` (`class_id`),
    CONSTRAINT `FK_student_class_student` FOREIGN KEY (`student_id`)
REFERENCES `student` (`student_id`)
);
```

```
-- 表 studentdb.student_class 的資料
INSERT INTO `student_class` (`student_id`, `class_id`, `score`) VALUES
    ('u001', '101-07', 80),
    ('u001', '275-08', 75),
    ('u001', '579-02', 68),
    ('u002', '101-07', 72),
    ('u002', '275-08', 77),
    ('u003', '101-07', 62),
    ('u003', '579-02', 55);
```

## Lab6: 新生資料新增，使用 SQL command

新生：蔡依林基本資料

```
-- 表 studentdb.student 的資料
INSERT INTO `student` (`student_id`, `name`, `phone`) VALUES
    ('u004', '蔡依林', '0938875178');
```

導師：

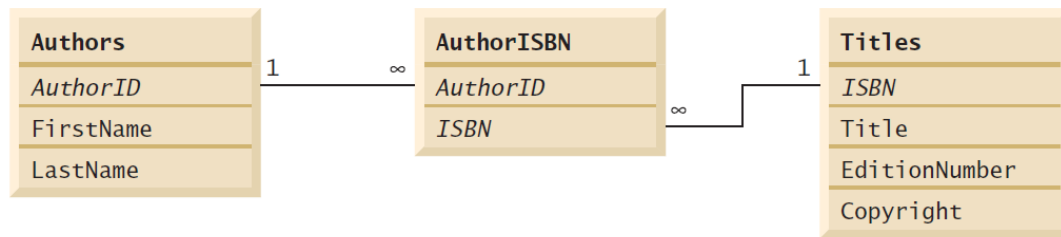
```
-- 表 studentdb.student_advisor 的資料
INSERT INTO `student_advisor` (`Student_id`, `Teacher_id`) VALUES
    ('u004', 't01');
```

修課資料

```
-- 表 studentdb.student_class 的資料
INSERT INTO `student_class` (`student_id`, `class_id`, `score`) VALUES
    ('u004', '101-07', 95),
    ('u004', '275-08', 88);
```



## Lab7 補充資料: Deitel 課本的關聯資料庫範例



AuthorID	FirstName	LastName
1	Paul	Deitel
2	Harvey	Deitel
3	Abbey	Deitel
4	Dan	Quirk
5	Michael	Morgano

AuthorID	ISBN
1	0132151006
2	0132151006
3	0132151006
1	0133807800
2	0133807800
1	0132575655
2	0132575655
1	013299044X
2	013299044X
1	0132990601
2	0132990601
3	0132990601
1	0133406954
2	0133406954
3	0133406954
1	0133379337

ISBN	Title	EditionNumber	Copyright
0132151006	Internet & World Wide Web How to Program	5	2012
0133807800	Java How to Program	10	2015
0132575655	Java How to Program, Late Objects Version	10	2015
013299044X	C How to Program	7	2013
0132990601	Simply Visual Basic 2010	4	2013
0133406954	Visual Basic 2012 How to Program	6	2014
0133379337	Visual C# 2012 How to Program	5	2014
0136151574	Visual C++ 2008 How to Program	2	2008
0133378713	C++ How to Program	9	2014
0133570924	Android How to Program	2	2015
0133570924	Android for Programmers: An App-Driven Approach, Volume 1	2	2014
0132121360	Android for Programmers: An App-Driven Approach	1	2012

--這是說明 (就像 Java 的 //) 以下的命令，大小寫皆可行，使用中文也可通

```
DROP TABLE authorISBN;
```

```
DROP TABLE titles;
```

```
DROP TABLE authors;
```

```
CREATE TABLE authors (  
    authorID INT NOT NULL GENERATED ALWAYS AS IDENTITY,  
    firstName varchar (20) NOT NULL,  
    lastName varchar (30) NOT NULL,  
    PRIMARY KEY (authorID)  
);
```

```
CREATE TABLE authors (  
    authorID INT NOT NULL,  
    firstName varchar (20) NOT NULL,  
    lastName varchar (30) NOT NULL,  
    PRIMARY KEY (authorID)  
);
```

```
CREATE TABLE titles (  
    isbn varchar (20) NOT NULL,  
    title varchar (100) NOT NULL,  
    editionNumber INT NOT NULL,  
    copyright varchar (4) NOT NULL,  
    PRIMARY KEY (isbn)  
);
```

```
CREATE TABLE authorISBN (  
    authorID INT NOT NULL,  
    isbn varchar (20) NOT NULL,  
    FOREIGN KEY (authorID) REFERENCES authors (authorID),  
    FOREIGN KEY (isbn) REFERENCES titles (isbn)  
);
```

```
INSERT INTO authors (firstName, lastName)
```

VALUES

```
('Paul','Deitel'),
('Harvey','Deitel'),
('Abbey','Deitel'),
('Michael','Morgano'),
('Eric','Kern');
```

INSERT INTO authors (authorID,firstName, lastName)

VALUES

```
(1,'Paul','Deitel'),
(2,'Harvey','Deitel'),
(3,'Abbey','Deitel'),
(4,'Michael','Morgano'),
(5,'Eric','Kern');
```

INSERT INTO titles (isbn,title,editionNumber,

copyright)

VALUES

```
('0132152134','Visual Basic 2010 How to Program',5,'2011'),
('0132151421','Visual C# 2010 How to Program',4,'2011'),
('0132575663','Java How to Program',9,'2012'),
('0132662361','C++ How to Program',8,'2012'),
('0132404168','C How to Program',6,'2010'),
('013705842X','iPhone for Programmers: An App-Driven Approach',1,'2010'),
('0132121360','Android for Programmers: An App-Driven Approach',1,'2012');
```

INSERT INTO authorISBN (authorID,isbn)

VALUES

```
(1,'0132152134'),
(2,'0132152134'),
(1,'0132151421'),
(2,'0132151421'),
(1,'0132575663'),
(2,'0132575663'),
(1,'0132662361'),
(2,'0132662361'),
```

```
(1,'0132404168'),
(2,'0132404168'),
(1,'013705842X'),
(2,'013705842X'),
(3,'013705842X'),
(4,'013705842X'),
(5,'013705842X'),
(1,'0132121360'),
(2,'0132121360'),
(3,'0132121360'),
(4,'0132121360');
```

```
/*
--insert into authors (authorId, name) values (40, '黃大大');
--delete from authors where authorid = 40;
--update authors set name = '李曉同' where authorid = 10;
--select * from AUTHORS;
```

--合併兩個表

```
select authors.AUTHORID, authors.FIRSTNAME, authors.LASTNAME,
authorisbn.ISBN from AUTHORS inner join AUTHORISBN on authors.AUTHORID
= authorisbn.AUTHORID;
```

```
select titles.ISBN, title, AUTHORID from titles inner join AUTHORISBN on
titles.ISBN = authorisbn.ISBN;
```

--合併三個表

```
select * from titles inner join AUTHORISBN on titles.ISBN = authorisbn.ISBN inner
join authors on authorisbn.AUTHORID = authors.AUTHORID ;
*/
```