



# 第8章 SQL結構化查詢語言

- 8-1 SQL結構化查詢語言的基礎
- 8-2 資料庫的實體資料模型
- 8-3 資料定義語言
- 8-4 資料操作語言
- 8-5 資料控制語言
- 8-6 產生SQL Server指令碼精靈





# 8-1 SQL語言的基礎

- 8-1-1 SQL結構化查詢語言
- 8-1-2 關聯式資料庫管理系統的查詢最佳化





## 8-1-1 SQL結構化查詢語言-說明

- 「SQL」（Structured Query Language）的全名是結構化查詢語言，筆者簡稱為SQL語言，SQL語言是「ANSI」（American National Standards Institute）制定的標準資料庫語言，其版本分為：
  - 1989年的ANSI-SQL 89
  - 1992年制定的ANSI-SQL 92，也稱為SQL 2，這是目前關聯式資料庫的標準語言
  - 最新版ANSI-SQL 99稱為SQL 3，適用在物件關聯或導向式資料庫的SQL語言。



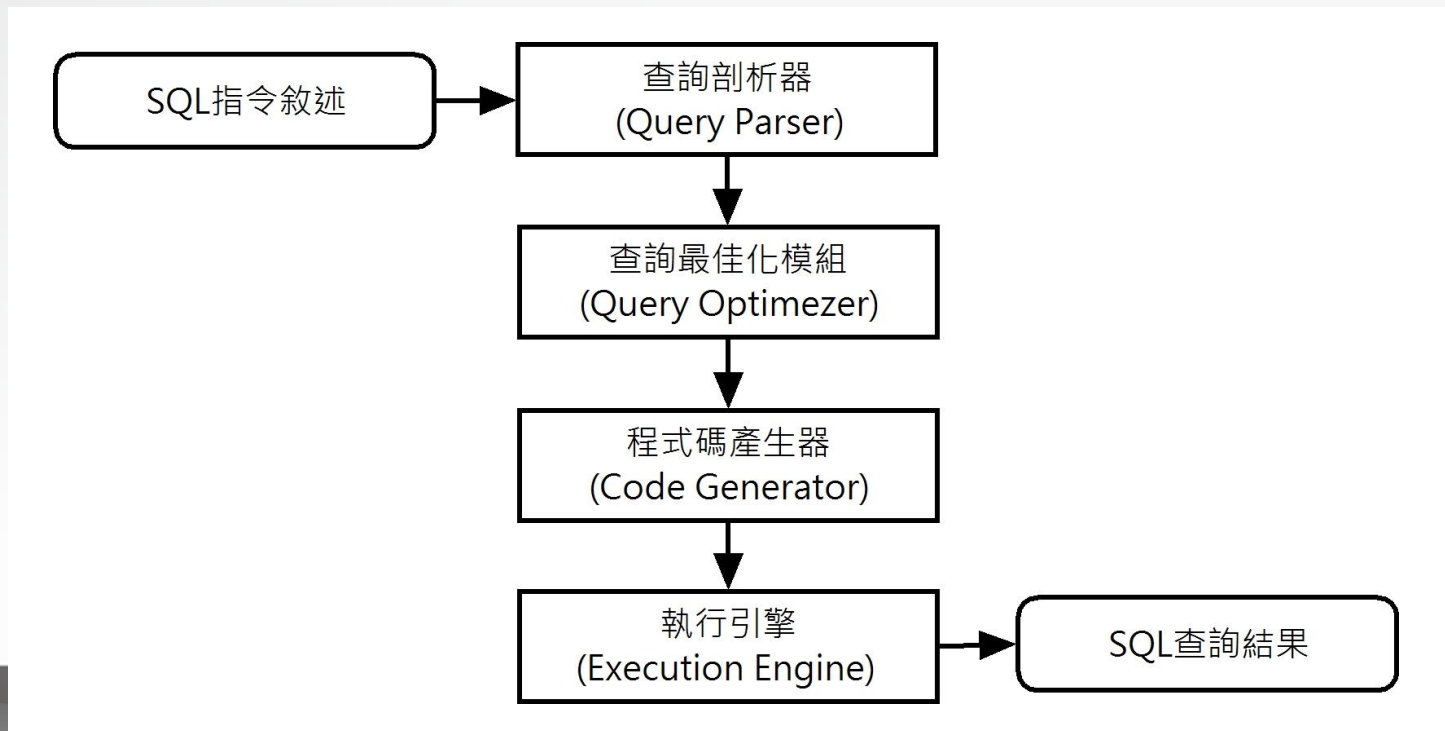
## 8-1-1 SQL結構化查詢語言-種類

- SQL語言依指令功能，可以分成三類，如下所示：
  - **資料定義語言DDL (Data Definition Language)**：建立資料表、視界和索引等的SQL指令。
  - **資料操作語言DML (Data Manipulation Language)**：資料表記錄插入、刪除、更新和查詢指令。
  - **資料控制語言DCL (Data Control Language)**：資料庫安全管理的權限設定指令。

## 8-1-2 關聯式資料庫管理系統的查詢最佳化-執行順序圖例



- SQL指令需要轉換成低階機器語言指令來執行查詢，其轉換的步驟即查詢處理模組（Query Processor）子系統的執行順序，如下圖所示：



## 8-1-2 關聯式資料庫管理系統的查詢最佳化- SQL查詢的步驟



- 第一步：在查詢剖析器檢查SQL語言的語法是否正確，且驗證屬性和關聯表名稱是否存在。
- 第二步：在查詢最佳化模組將它轉換成中間格式關聯式代數建立的「查詢樹」（Query Tree），然後執行最佳化處理產生「執行計劃」（Execution Plan）。
- 第三步：使用RDBMS提供的選擇、合併、投影等關聯代數運算子的演算法，從執行計劃產生所需的程式碼。
- 第四步：使用直譯或編譯方式將程式碼轉換成機器語言後，就可以執行SQL指令產生查詢結果。

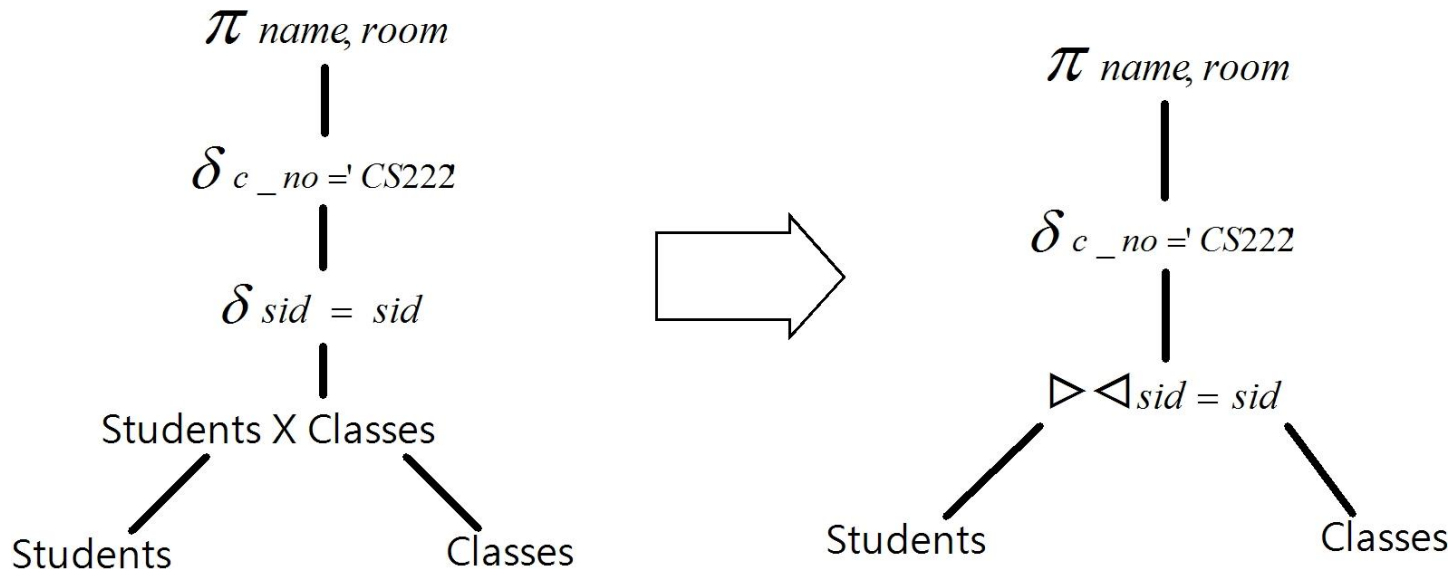
## 8-1-2 關聯式資料庫管理系統的查詢最佳化- 查詢樹(Query Tree)



- 查詢樹（Query Tree）是一棵關聯式代數運算式的樹狀結構，查詢樹的節點規則，如下所示：
  - 關聯表是樹的葉節點（Leaf Nodes）。
  - 關聯式代數運算子是中間節點（Internal Nodes）。



## 8-1-2 關聯式資料庫管理系統的查詢最佳化- 查詢樹圖例



```
SELECT Students.name, Classes.room  
FROM Students, Classes  
WHERE Students.sid = Classes.sid  
and Classes.c_no = 'CS222'
```



## 8-1-2 關聯式資料庫管理系統的查詢最佳化- 查詢最佳化模組



- 查詢最佳化模組（Query Optimizer）的功能是在最佳化查詢樹，讓查詢樹執行的更有效率。
  - 最佳化的作法有很多種，最簡單的方式是將查詢樹的節點轉換成更有效率且相等功能的節點，這些最佳化規則稱為啟發式查詢最佳化（Heuristic Query Optimization）技術。

## 8-1-2 關聯式資料庫管理系統的查詢最佳化- 查詢最佳化技術1



- 啟發式查詢最佳化（**Heuristic Query Optimization**）：這是規則基礎的最佳化，使用關聯式代數已知的轉換規則產生最有效率的執行計劃，例如：將卡笛生乘積運算和之後的選擇運算以合併運算來取代。
- 造句法式查詢最佳化（**Syntactical Query Optimization**）：以查詢樹自行造句找出所有同等功能的可能關聯式代數運算式，其最佳化的依據主要在評估是否有索引可以改進執行效率，以便找出最佳的執行計劃。

## 8-1-2 關聯式資料庫管理系統的查詢最佳化- 查詢最佳化技術2



- 成本基礎式查詢最佳化（**Cost-Based Query Optimization**）：使用RDBMS各種資訊評估使用那一種順序執行關聯式代數運算式擁有最低的成本，以找出最有效率的存取方式，這是SQL Server和目前大多數RDBMS採用的最佳化技術。
- 語意式查詢最佳化（**Semantic Query Optimization**）：語意式查詢最佳化需要查詢最佳化模組了解資料庫綱要，可以自行依所知的系統限制來簡化或刪除查詢結果是空集合的部分查詢，而且可以自行評估是否使用索引來加速資料存取。

## 8-1-2 關聯式資料庫管理系統的查詢最佳化- 執行計劃



- 執行計劃（**Execution Plan**）是從已經最佳化的查詢樹轉換而得的一個執行策略。執行計劃包含如何存取每一個關聯表的資料存取方式。
  - 例如：關聯表使用那一個屬性排序，或是否存在其他屬性的索引。



## 8-1-2 關聯式資料庫管理系統的查詢最佳化- 執行計劃演算法

- 選擇運算：使用線性搜尋（Linear Search）、二元搜尋（Binary Search）或雜湊表搜尋法（Hash Table Search）來找尋值組，在索引部分可以使用叢集索引（Clustering Index）或B-樹演算法。
- 合併運算：使用巢狀迴圈合併（Nested-loop Join）、單迴圈合併（Single-loop Join）、排序合併（Sort Merge Join）和雜湊合併（Hashing Merge Join）等多種合併演算法。
- 投影運算：沒有特別的演算法，因為只是刪除一些關聯表的屬性。



## 8-2 資料庫的實體資料模型- Students資料表

- 當完成School學校資料庫設計後，我們就可以依據資料庫設計來建立實體資料模型（Physical Data Model），然後在資料庫使用SQL指令新增所需的資料表。

### Students資料表

欄位名稱	資料類型	長度	欄位說明
<u>sid</u>	CHAR	4	學生的學號，主鍵
name	VARCHAR	12	姓名
tel	VARCHAR	15	電話
birthday	DATE	N/A	生日



## 8-2 資料庫的實體資料模型- Instructors與Courses資料表

### Instructors資料表

欄位名稱	資料類型	長度	欄位說明
<u>eid</u>	CHAR	4	講師的員工編號，主鍵
name	VARCHAR	12	講師姓名
rank	VARCHAR	10	講師的等級
department	VARCHAR	5	所屬科系

### Courses資料表

欄位名稱	資料類型	長度	欄位說明
<u>c_no</u>	CHAR	5	課程編號，主鍵
title	VARCHAR	30	課程名稱
credits	INT	N/A	學分數



## 8-2 資料庫的實體資料模型-Classes資料表

### Classes資料表

欄位名稱	資料類型	長度	欄位說明
<u>eid</u>	CHAR	4	講師的員工編號，參考 Instructors 資料表的 eid 欄位
<u>sid</u>	CHAR	4	學生學號，參考 Students 資料表的 sid 欄位
<u>c_no</u>	CHAR	5	課程編號，參考 Courses 資料表的 c_no 欄位
time	TIME	N/A	上課時間
room	VARCHAR	8	教室編號
grade	FLOAT	N/A	這門課程的成績





## 8-3 資料定義語言

- 8-3-1 SQL資料定義語言的基礎
- 8-3-2 CREATE TABLE新增關聯表
- 8-3-3 ALTER TABLE更改關聯表綱要
- 8-3-4 DROP TABLE刪除整個關聯表
- 8-3-5 TRUNCATE TABLE刪除關聯表內容
- 8-3-6 CREATE INDEX建立索引
- 8-3-7 DROP INDEX刪除索引





## 8-3-1 SQL資料定義語言的基礎

- SQL資料定義語言（DDL）可以建立關聯式資料庫綱要，也就是在資料庫建立每一個關聯表綱要、視界、索引和新增完整性限制條件。DDL指令可以分成數種，如下所示：
  - 建立資料表指令：CREATE TABLE指令可以建立基底關聯表，同時還可以新增關聯表的完整性限制條件，在這一節主要說明的是DDL指令。
  - 建立視界指令：建立視界（Views）的指令是CREATE VIEW，詳細說明請參閱〈第11章：視界與資料庫程式設計〉。
  - 建立索引指令：建立資料表索引的DDL指令，也將在本節說明。



## 8-3-2 CREATE TABLE 新增關聯表- CREATE TABLE 語法

```
CREATE TABLE table_name (  
    column1 datatype [ NOT NULL | NULL ] [ DEFAULT value1 ],  
    column2 datatype [ NOT NULL | NULL ] [ DEFAULT value2 ],  
    column3 datatype [ NOT NULL | NULL ] [ DEFAULT value3 ],  
    .....  
    columnN datatype [ NOT NULL | NULL ] [ DEFAULT valueN ]  
    [ , PRIMARY KEY (column_list) ]  
    [ , FOREIGN KEY (column_list) REFERENCES table_name  
      (column_list)  
    [ , CHECK (expr) ]  
)
```



## 8-3-2 CREATE TABLE新增關聯表-語法說明

- **NOT NULL | NULL**：欄位值是否可以為虛值，如果沒有指明，預設是NULL，可以是空值。
- **DEFAULT**：指定欄位的預設值，如果欄位沒有輸入資料，預設是填入之後的value1~N值。
- **PRIMARY KEY ()**：指定資料表的主鍵，括號內是主鍵的欄位，如果是複合鍵，欄位請使用逗號分隔。
- **FOREIGN KEY ()/REFERENCES ()**：指定資料表的外來鍵，括號為外來鍵的欄位清單，REFERENCE是參考的資料表，括號是參考資料表的主鍵。
- **CHECK ()**：設定欄位的限制條件，在括號內是條件的運算式。



## 8-3-2 CREATE TABLE新增關聯表- 建立Students資料表

建立Students資料表：Ch8\_3\_2\_01.sql

- 首先在MySchool資料庫建立Students資料表，其SQL指令如下所示：

```
CREATE TABLE Students (  
    sid    char(4)    NOT NULL ,  
    name   varchar(12) NOT NULL ,  
    tel    varchar(15),  
    birthday date ,  
    PRIMARY KEY (sid)  
)  
GO
```



## 8-3-2 CREATE TABLE新增關聯表- 建立Instructors資料表

建立Instructors資料表：Ch8\_3\_2\_02.sql

- 在MySchool資料庫建立Instructors資料表的SQL指令，主鍵是eid欄位，如下所示：

```
CREATE TABLE Instructors (  
    eid    char(4)    NOT NULL,  
    name   varchar(12) NOT NULL,  
    rank   varchar(10),  
    department varchar(5),  
    PRIMARY KEY (eid)  
)
```



## 8-3-2 CREATE TABLE新增關聯表- 建立Courses資料表

建立Courses資料表：Ch8\_3\_2\_03.sql

- 在MySchool資料庫建立Courses資料表的SQL指令，主鍵是c\_no欄位，如下所示：

```
CREATE TABLE Courses (  
    c_no char(5) NOT NULL,  
    title varchar(30) NOT NULL,  
    credits int DEFAULT 3,  
    PRIMARY KEY (c_no)  
)
```



## 8-3-2 CREATE TABLE 新增關聯表- 建立Classes資料表

### 建立Classes資料表：Ch8\_3\_2\_04.sql

- 在MySchool資料庫建立Classes資料表的SQL指令，如下所示：

```
CREATE TABLE Classes (  
    eid char(4) NOT NULL,  
    sid char(4) NOT NULL,  
    c_no char(5) NOT NULL,  
    time time,  
    room varchar(8),  
    grade float,  
    PRIMARY KEY (eid, sid, c_no),  
    FOREIGN KEY (eid) REFERENCES Instructors (eid),  
    FOREIGN KEY (sid) REFERENCES Students (sid),  
    FOREIGN KEY (c_no) REFERENCES Courses (c_no)
```

)





## 8-3-3 ALTER TABLE更改關聯表綱要- 新增/刪除欄位(語法)

### 新增/刪除資料表欄位

- 如果資料表的欄位需要增減，並不需要重新建立資料表，可以使用**ALTER TABLE**指令來新增/刪除欄位，其基本語法如下所示：

**ALTER TABLE table\_name**

**ADD column\_name datatype**

**DROP COLUMN column\_name**



## 8-3-3 ALTER TABLE更改關聯表綱要- 新增/刪除欄位(範例)

新增資料表欄位：Ch8\_3\_3\_01.sql

- 在Students資料表新增SSN欄位的身份證字號，資料類型是char(10)，其SQL指令如下所示：

```
ALTER TABLE Students  
ADD SSN char(10)
```

刪除資料表欄位：Ch8\_3\_3\_02.sql

- 在Students資料表刪除欄位SSN，其SQL指令如下所示：

```
ALTER TABLE Students  
DROP COLUMN SSN
```



## 8-3-3 ALTER TABLE更改關聯表綱要- 新增主鍵(語法)

### 新增/刪除資料表的主鍵

- 如果在建立資料表時沒有指定主鍵，可以使用 **ALTER TABLE** 指令新增資料表的主鍵，其基本語法如下所示：

**ALTER TABLE table\_name**

**ADD PRIMARY KEY (column\_list)**

**DROP PRIMARY KEY**



## 8-3-3 ALTER TABLE更改關聯表綱要- 新增主鍵(範例)

新增主鍵：Ch8\_3\_3\_04.sql

- 請先執行SQL指令檔名稱：Ch8\_3\_3\_03.sql建立Employees資料表，但沒有指定主鍵，我們可以使用Ch8\_3\_3\_04.sql的ALTER TABLE指令新增Employees的主鍵SSN，如下所示：

```
ALTER TABLE Employees  
ADD PRIMARY KEY (SSN)
```



## 8-3-3 ALTER TABLE更改關聯表綱要- 新增外來鍵(語法)

### 新增/刪除資料表的外來鍵

- 如果在建立資料表時沒有指定外來鍵，可以使用 **ALTER TABLE** 指令新增資料表的外來鍵，也就是建立關聯表間的關聯性（**Relationships**），其基本語法如下所示：

**ALTER TABLE table\_name**

**ADD CONSTRAINT constraint\_name**

**FOREIGN KEY (column\_list) REFERENCES table\_name  
(column\_list)**

**DROP CONSTRAINT constraint\_name**



## 8-3-3 ALTER TABLE更改關聯表綱要- 新增外來鍵(範例1)

新增外來鍵：Ch8\_3\_3\_05.sql

- 繼續使用之前建立的Employees資料表為例，使用ALTER TABLE指令新增Employees資料表的外來鍵eid欄位，它是參考到Instructors資料表的eid欄位，其SQL指令如下所示：

```
ALTER TABLE Employees
```

```
ADD CONSTRAINT constraint_eid
```

```
FOREIGN KEY (eid) REFERENCES Instructors (eid)
```



## 8-3-3 ALTER TABLE更改關聯表綱要- 新增外來鍵(範例2)

刪除外來鍵：Ch8\_3\_3\_06.sql

- 在Employees資料表刪除外來鍵，也就是刪除名為constraint\_eid的限制條件，其SQL指令如下所示：

```
ALTER TABLE Employees
```

```
DROP CONSTRAINT constraint_eid
```



## 8-3-3 ALTER TABLE更改關聯表綱要- 新增/刪除欄位的限制條件(語法)

### 新增/刪除欄位的限制條件

- ALTER TABLE指令可以新增/刪除資料表欄位的限制條件，其基本語法如下所示：

ALTER TABLE table\_name

ADD CONSTRAINT constraint\_name

DEFAULT value FOR column\_name | CHECK (expr)

DROP CONSTRAINT constraint\_name





## 8-3-3 ALTER TABLE更改關聯表綱要- 新增/刪除欄位的限制條件(範例1)

新增欄位預設值：Ch8\_3\_3\_07.sql

- 在Employees資料表新增name欄位的預設值，其SQL指令如下所示：

```
ALTER TABLE Employees  
ADD CONSTRAINT default_name  
DEFAULT '陳會安' FOR name
```



## 8-3-3 ALTER TABLE更改關聯表綱要- 新增/刪除欄位的限制條件(範例2)

新增欄位的限制條件：Ch8\_3\_3\_08.sql

- 在Students學生資料表新增生日birthday欄位的限制條件，其SQL指令如下所示：

```
ALTER TABLE Students
```

```
ADD CONSTRAINT check_birthday
```

```
CHECK (birthday >= '1950-1-1' and birthday <= '2000-12-31')
```



## 8-3-4 DROP TABLE刪除整個關聯表

- 對於資料庫已經存在的基底關聯表，可以使用 **DROP TABLE** 指令刪除指定的資料表，刪除的範圍包含資料表索引、記錄和視界，其基本語法如下所示：

**DROP TABLE table\_name**

- 上述語法是從資料庫刪除table\_name的資料表。

刪除資料表：Ch8\_3\_4\_01.sql

- 刪除Employees資料表的SQL指令，如下所示：

**DROP TABLE Employees**



## 8-3-5 TRUNCAT TABLE刪除關聯表內容

- 資料庫已經存在的基底關聯表，如果想刪除資料表的記錄，但是保留關聯表綱要，可以使用 **TRUNCAT TABLE** 指令來刪除指定資料表的內容，其基本語法如下所示：

**TRUNCATE TABLE table\_name**

刪除資料表的內容：Ch8\_3\_5\_01.sql

- 刪除Classes資料表內容，但是保留關聯表綱要，其SQL指令如下所示：

**TRUNCATE TABLE Classes**



## 8-3-6 CREATE INDEX 建立索引-說明

- 以單層有序索引（**Single Level Ordered Index**）來說，使用主鍵排序建立的索引稱為主索引（**Primary Index**）。非主鍵排序的索引稱為叢集索引（**Clustering Index**），其他非排序欄位稱為次索引（**Secondary Index**），也稱為非叢集索引（**Non-clustering Index**）。
- 一個基底關聯表只能擁有一個主索引或叢集索引來決定記錄的排序方式，如果還有其他索引一定是次索引，在一個基底關聯表可以擁有多個次索引。
- **SQL Server**可以建立叢集和非叢集索引，當在資料表指定主鍵時，預設建立此欄位的叢集索引（當然我們也可稱它為主索引），其他欄位的索引稱為非叢集索引（即次索引）。



## 8-3-6 CREATE INDEX 建立索引-語法

- DDL的CREATE INDEX指令可以建立基底關聯表的索引，其基本語法如下所示：

```
CREATE [ UNIQUE ] INDEX index_name  
    ON table_name (column_name[(length)],... ) [ ASC|DESC ]
```

- 上述語法可以建立SQL Server資料表的非叢集索引，UNIQUE是指索引值為唯一，表示不能有2筆記錄擁有相同索引值，index\_name是索引名稱，column\_name欄位如果不只一個，請使用逗號分隔。length欄位是長度，可以使用部分欄位值來建立索引資料，ASC是由小到大排序，DESC是由大到小。

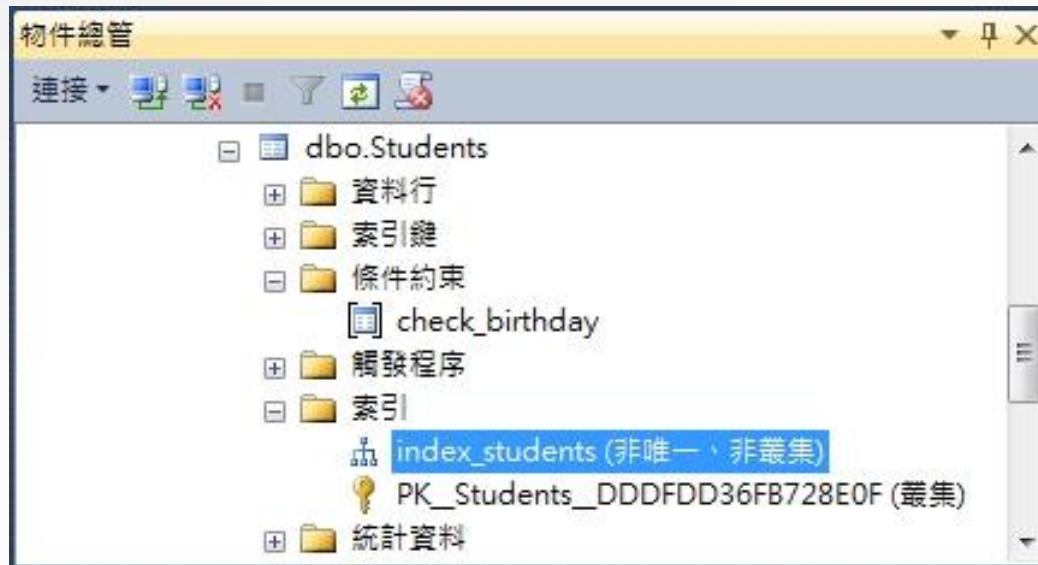


## 8-3-6 CREATE INDEX 建立索引|-範例

建立資料表的索引：Ch8\_3\_6\_01.sql

- 在Students學生資料表新增欄位name的索引資料index\_students，其SQL指令如下所示：

CREATE INDEX index\_students ON Students(name)





## 8-3-7 DROP INDEX刪除索引

- DDL的DROP INDEX指令可以刪除基底關聯表的索引資料，其基本語法如下所示：

**DROP INDEX table\_name.index\_name**

- 上述table\_name是資料表名稱，index\_name是索引名稱。

刪除資料表的索引：Ch8\_3\_7\_01.sql

- 在Students學生資料表刪除索引資料的SQL指令，如下所示：

**DROP INDEX Students.index\_students**





## 8-4 資料操作語言

- 8-4-1 INSERT新增記錄
- 8-4-2 UPDATE更新記錄
- 8-4-3 DELETE刪除記錄
- 8-4-4 Management Studio編輯資料表





## 8-4 資料操作語言

- SQL的資料操作語言（DML）可以新增、刪除和更新資料表的記錄，換句話說，DDL是建立資料庫綱要，DML是建立資料庫副本（Instance）。
- SQL語言的資料表操作指令一共有3個，如下表所示：
  - **INSERT**：在資料表插入一筆新記錄
  - **UPDATE**：更新資料表的記錄，這些記錄是已經存在的記錄
  - **DELETE**：刪除資料表的記錄



## 8-4-1 INSERT新增記錄-語法

- SQL語言的INSERT指令可以新增一筆記錄到資料表，INSERT指令的基本語法，如下所示：

```
INSERT INTO table (column1,column2,...) VALUES  
( 'value1', 'value2 ', ...)
```

- 上述SQL指令的table為新增記錄的資料表名稱，column1~n是資料表內的欄位名稱，不需全部欄位，不過需要包含所有非虛值欄位，value1~n是對應的欄位值。



## 8-4-1 INSERT新增記錄-注意事項

- INSERT指令的注意事項，如下所示：
  - 不論是欄位或值的清單，都需要使用逗號分隔。
  - 在INSERT指令VALUES的值中，數值不用單引號包圍，字元與日期/時間需要單引號括起。
  - INSERT指令的欄位名稱清單，並不需要和資料表定義的欄位數目或順序相同，只需列出需要值的欄位，不過括號內的欄位名稱順序要和VALUES值的順序相同。



## 8-4-1 INSERT新增記錄-範例

### 新增記錄：Ch8\_4\_1\_01.sql

- 在Students、Instructors、Courses和Classes四個資料表新增測試記錄的SQL指令，如下所示：

```
INSERT INTO Students
```

```
VALUES ('S001','陳小安','02-22222222','1968/09/12')
```

```
.....
```

```
INSERT INTO Courses
```

```
VALUES ('CS101','計算機概論',4)
```

```
.....
```

```
INSERT INTO Instructors
```

```
VALUES ('E001','陳慶新','教授','CS')
```

```
.....
```

```
INSERT INTO Classes
```

```
VALUES ('E001','S001','CS101','12:00:00','180-M',85)
```

```
.....
```



## 8-4-2 UPDATE更新記錄-語法

- SQL語言的UPDATE指令可以將資料表符合條件的記錄，更新指定欄位的內容，基本語法如下所示：

```
UPDATE table SET column1 = 'value1' WHERE  
conditions
```

- table是資料表，SET子句column1是資料表的欄位名稱，不用全部只需更新欄位，value1是更新欄位值。
- 更新的欄位不只一個請使用逗號分隔，如下：

```
UPDATE table SET column1 = 'value1', column2 =  
'value2' WHERE conditions
```



## 8-4-2 UPDATE更新記錄-注意事項與範例

- UPDATE指令的注意事項，如下所示：
  - WHERE子句是UPDATE指令的必要元素，如果沒有此條件，資料表所有記錄欄位都會被更新。
  - 更新欄位值如果為數值不用單引號包圍，字元與日期/時間需要使用單引號包圍。

更新記錄：Ch8\_4\_2\_01.sql

- 在資料表Students更改學號S001的生日和姓名，其SQL指令如下所示：

```
UPDATE Students
```

```
SET birthday='1967-09-03' , name='陳會安'
```

```
WHERE sid = 'S001'
```



## 8-4-3 DELETE刪除記錄-語法

- SQL語言的DELETE指令可以將資料表符合條件的記錄刪除掉，DELETE指令的基本語法，如下所示：

**DELETE FROM table WHERE conditions**

- table是資料表，WHERE子句conditions為刪除記錄的條件，口語來說是：「將符合conditions條件的記錄刪除掉」。





## 8-4-3 DELETE刪除記錄-注意事項與範例

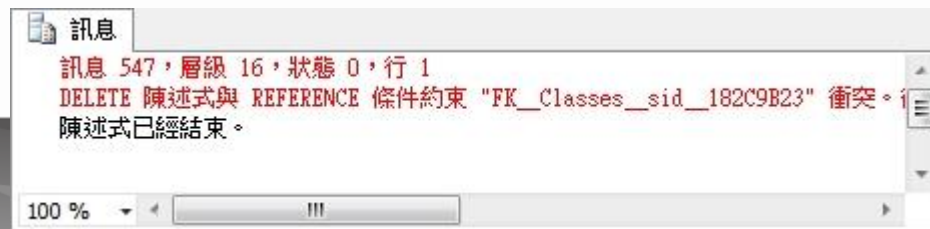
### ■ DELETE指令的注意事項，如下所示：

- WHERE子句是DELETE指令的必要元素，如果沒有此條件，資料表內的所有記錄都會被刪除掉。
- WHERE條件可以使用=、<>、>、<=和>=等運算子。
- WHERE條件如果不只一個條件，可以使用邏輯運算子and或or連結。

刪除記錄：Ch8\_4\_3\_01.sql

### ■ 在Students資料表刪除學號S001記錄，其SQL指令如下所示：

**DELETE FROM Students WHERE sid = 'S001'**





## 8-4-4 Management Studio編輯資料表

- SQL Server在建立MySchool資料庫和使用SQL指令新增四個資料表後，就可以使用Management Studio在資料表新增測試的記錄資料。

The screenshot shows a window titled 'JOE-PC\SQLEXPRES... | - dbo.Students'. It displays a table with the following columns: sid, name, tel, and birthday. The table contains 5 rows of data, with the first row selected. A new row is being added at the bottom, indicated by an asterisk and NULL values.

	sid	name	tel	birthday
▶	S001	陳會安	02-22222222	1967-09-03
	S002	江小魚	03-33333333	1978-02-02
	S003	張三羊	04-44444444	1982-03-03
	S004	李四方	05-55555555	1981-04-04
	S005	陳允傑	02-22222222	1966-09-03
*	NULL	NULL	NULL	NULL



## 8-5 資料控制語言-說明

- SQL語言的資料控制語言（Data Control Language，DCL）可以定義資料庫安全管理所需的規則，授予或撤回使用者的相關權限，讓使用者：
  - 只能執行授權的操作和處理擁有權限的資料。
  - 限制無法執行或取得沒有授權的操作或資料。



## 8-5 資料控制語言-GRANT指令

- GRANT指令可以授予資料庫使用者的權限，其基本語法如下所示：

```
GRANT { ALL [ PRIVILEGES ]  
      | privileges [ (column [ ,columnn]) ]  
      [,privileges...] }  
ON table_name  
TO user_name [,user_name ...]
```

- 上述語法可以在ON子句指定授予哪一個table\_name資料表的權限，在privileges權限的括號中可以指定擁有哪些欄位的權限。



## 8-5 資料控制語言-REVOKE指令

- REVOKE指令是對應GRANT指令，可以撤回某位使用者所授予的權限，其基本語法如下所示：

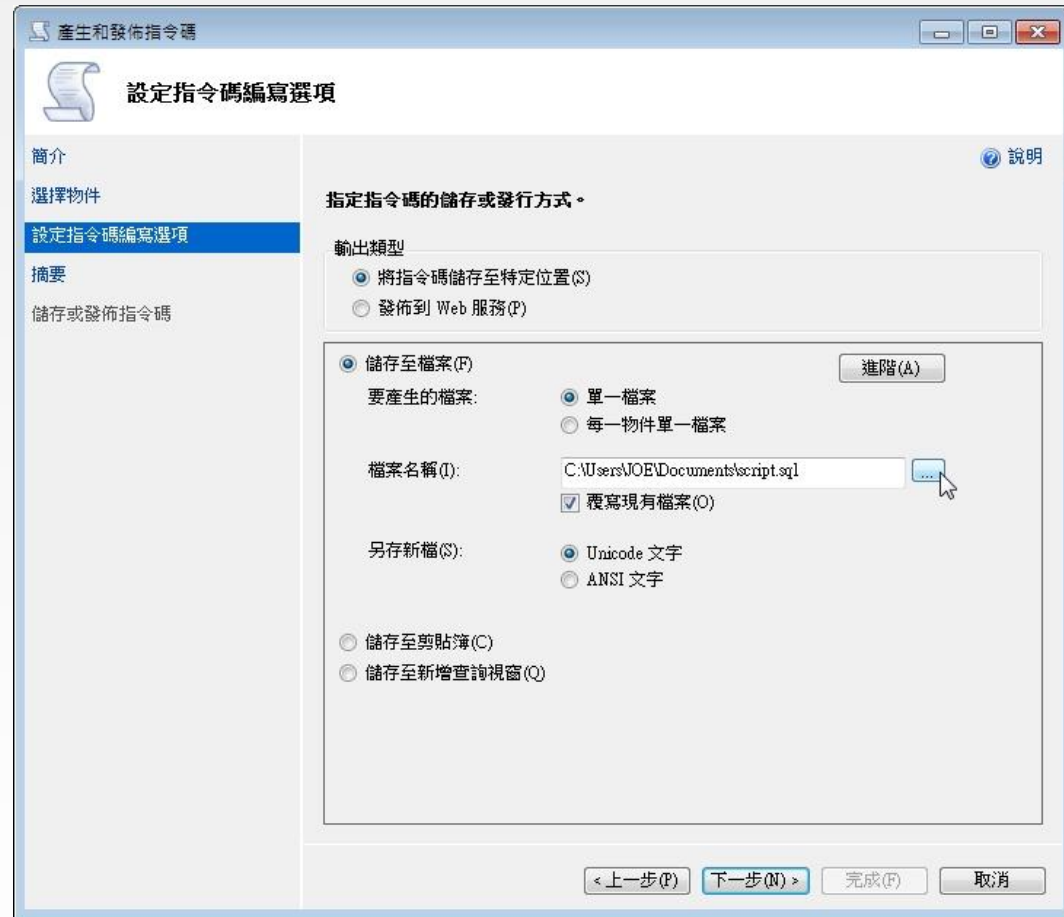
```
REVOKE { ALL [ PRIVILEGES ]  
        | privileges [ (column [ ,columnn]) ]  
        [,privileges...] }  
ON table_name  
FROM user_name [,user_name ...]
```

- 上述語法可以在ON子句指定撤回哪一個table\_name資料表的權限，在privileges權限的括號中可以指定撤回哪些欄位的權限。



## 8-6 產生SQL Server指令碼精靈

- Management Studio 提供指令碼精靈來自動產生資料庫綱要的SQL指令碼檔案，其主要目的是如果資料庫損壞時，可以使用這些產生的SQL指令檔來還原損壞的資料庫綱要。





End

---