

資料庫入門

資料庫與表格管理

鄭安翔

ansel_cheng@hotmail.com

課程大綱

1) 資料庫管理

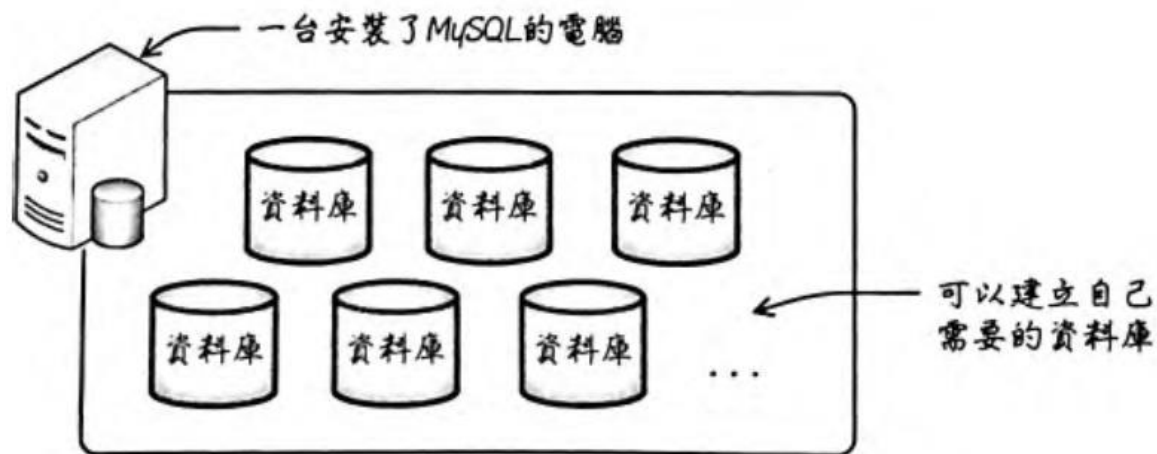
- 資料庫建立修改與刪除
- 資料庫字元集及排序
- 儲存引擎

2) 欄位資料型態

3) 表格管理

建立資料庫

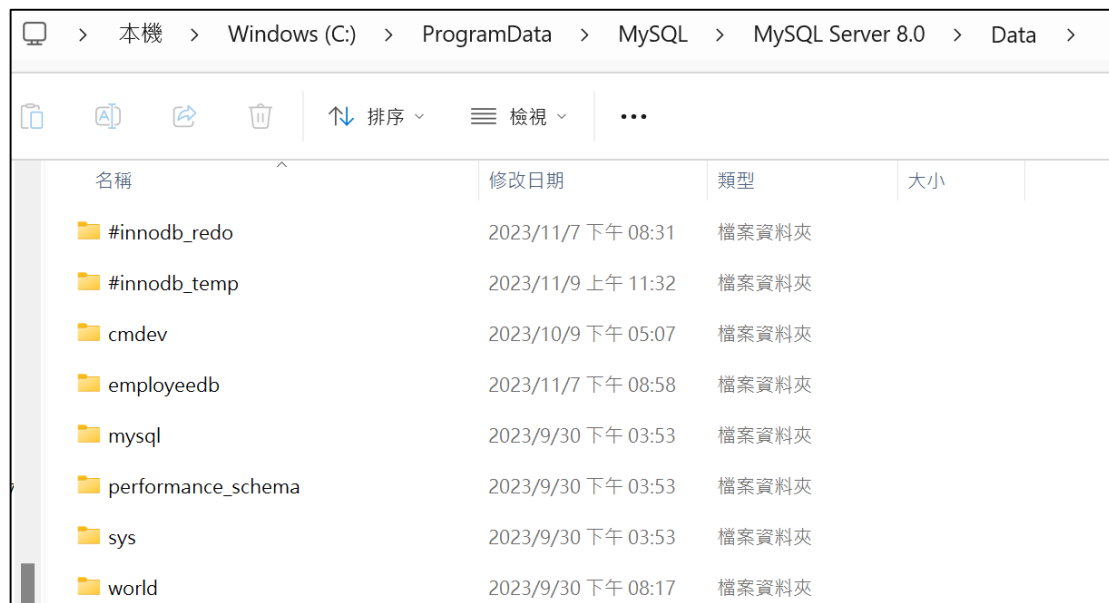
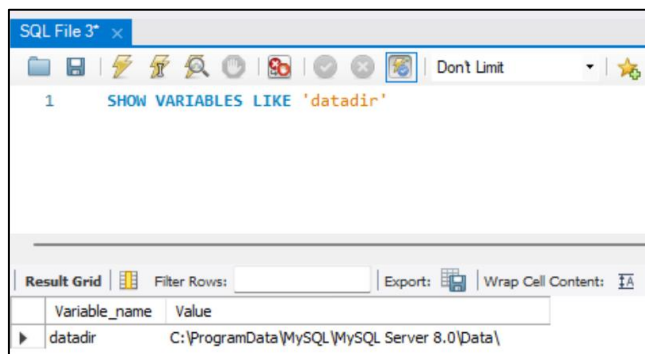
- 資料庫管理系統
 - 可以建立多個資料庫
 - MySQL資料庫數量沒有限制



MySQL儲存資料的資料夾

■ MySQL儲存資料的資料夾

□ SHOW VARIABLES LIKE 'datadir' 查詢資料夾

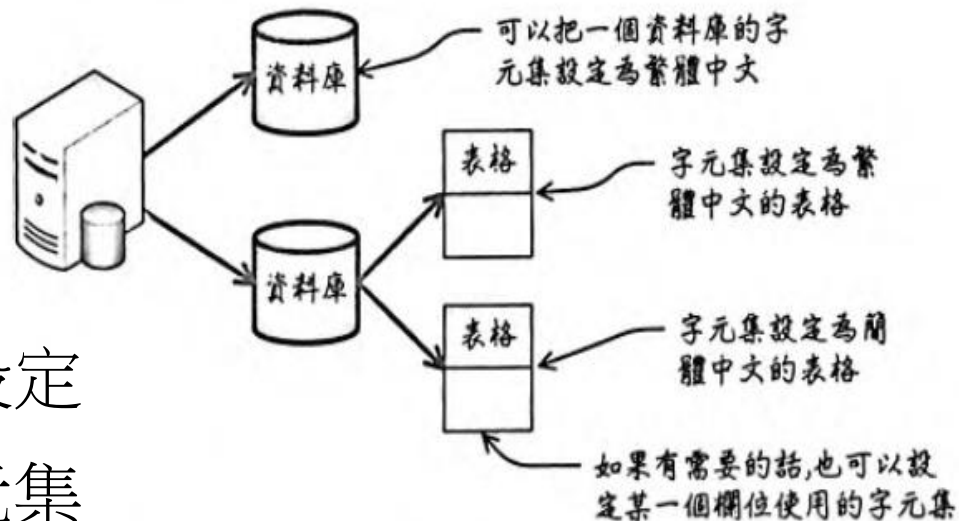


資料庫字元集

■ 字元集 character set

- ❑ 不同語言的文字編碼
- ❑ 可針對資料庫或表格設定
- ❑ 查詢資料庫支援之字元集

SHOW CHARACTER SET



文字資料排序規則 COLLATION

■ 文字資料排序規則 COLLATION

- 字元集中文字大小寫排序規則
- 可依照實際需要，搭配不同的Collation設定
 - bin : binary
 - cs : case-sensitive
 - ci : case-insensitive
- 查詢MySQL支援的Collation資訊

SHOW COLLATION

範例

■ 檢視MySQL支援的字元集及COLLATION

SQL File 3* x

1 • SHOW CHARACTER SET

Result Grid | Filter Rows: | Export: | Wrap Cell

	Charset	Description	Default collation	Maxlen
▶	armscii8	ARMSCII-8 Armenian	armscii8_general_ci	1
	ascii	US ASCII	ascii_general_ci	1
	big5	Big5 Traditional Chinese	big5_chinese_ci	2
	binary	Binary pseudo charset	binary	1
	cp1250	Windows Central European	cp1250_general_ci	1
	cp1251	Windows Cyrillic	cp1251_general_ci	1
	cp1256	Windows Arabic	cp1256_general_ci	1
	cp1257	Windows Baltic	cp1257_general_ci	1
	cp850	DOS West European	cp850_general_ci	1
	cp852	DOS Central European	cp852_general_ci	1
	cp866	DOS Russian	cp866_general_ci	1
	cp932	SJIS for Windows Japanese	cp932_japanese_ci	2
	dec8	DEC West European	dec8_swedish_ci	1
	ejcpms	UJIS for Windows Japanese	ejcpms_japanese_ci	3
	euokr	EUC-KR Korean	euokr_korean_ci	2
	gb18030	China National Standard G...	gb18030_chinese_ci	4
	gb2312	GB2312 Simplified Chinese	gb2312_chinese_ci	2
	gbk	GBK Simplified Chinese	gbk_chinese_ci	2
	geostd8	GEOSTD8 Georgian	geostd8_general_ci	1
	greek	ISO 8859-7 Greek	greek_general_ci	1
	hebrew	ISO 8859-8 Hebrew	hebrew_general_ci	1
	hp8	HP West European	hp8_english_ci	1
	keybcs2	DOS Kamenicky Czech-Slo...	keybcs2_general_ci	1
	koi8r	KOI8-R Relcom Russian	koi8r_general_ci	1
	koi8u	KOI8-U Ukrainian	koi8u_general_ci	1
	latin1	cp1252 West European	latin1_swedish_ci	1
	latin2	ISO 8859-2 Central Europ...	latin2_general_ci	1

SQL File 3* x

1 SHOW COLLATION

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	Collation	Charset	Id	Default	Compiled	Sortlen	Pad_attribute
▶	armscii8_bin	armscii8	64		Yes	1	PAD SPACE
	armscii8_general_ci	armscii8	32	Yes	Yes	1	PAD SPACE
	ascii_bin	ascii	65		Yes	1	PAD SPACE
	ascii_general_ci	ascii	11	Yes	Yes	1	PAD SPACE
	big5_bin	big5	84		Yes	1	PAD SPACE
	big5_chinese_ci	big5	1	Yes	Yes	1	PAD SPACE
	binary	binary	63	Yes	Yes	1	NO PAD
	cp1250_bin	cp1250	66		Yes	1	PAD SPACE
	cp1250_croatian_ci	cp1250	44		Yes	1	PAD SPACE
	cp1250_czech_cs	cp1250	34		Yes	2	PAD SPACE
	cp1250_general_ci	cp1250	26	Yes	Yes	1	PAD SPACE
	cp1250_polish_ci	cp1250	99		Yes	1	PAD SPACE
	cp1251_bin	cp1251	50		Yes	1	PAD SPACE
	cp1251_bulgarian_ci	cp1251	14		Yes	1	PAD SPACE
	cp1251_general_ci	cp1251	51	Yes	Yes	1	PAD SPACE
	cp1251_general_cs	cp1251	52		Yes	1	PAD SPACE
	cp1251_ukrainian_ci	cp1251	23		Yes	1	PAD SPACE
	cp1256_bin	cp1256	67		Yes	1	PAD SPACE
	cp1256_general_ci	cp1256	57	Yes	Yes	1	PAD SPACE
	cp1257_bin	cp1257	58		Yes	1	PAD SPACE
	cp1257_general_ci	cp1257	59	Yes	Yes	1	PAD SPACE
	cp1257_lithuanian_ci	cp1257	29		Yes	1	PAD SPACE
	cp850_bin	cp850	80		Yes	1	PAD SPACE
	cp850_general_ci	cp850	4	Yes	Yes	1	PAD SPACE
	cp852_bin	cp852	81		Yes	1	PAD SPACE
	cp852_general_ci	cp852	40	Yes	Yes	1	PAD SPACE
	cp866_bin	cp866	68		Yes	1	PAD SPACE

建立資料庫

■ 建立資料庫語法

CREATE DATABASE [IF NOT EXISTS] 資料庫名稱

[CHARACTER SET 字元集名稱]

[COLLATE Collation 名稱]

- 以資料庫名稱建立資料夾
- **IF NOT EXISTS** 可避免資料庫已存在的錯誤
- 預設字元集 **latin1**，預設Collation **latin1_swedish_ci**
- 只指定字元集，Collation為指定字元集預設collation
- 只指定Collation，字元集則為Collation所屬字元集

修改資料庫

- 修改資料庫語法

ALTER DATABASE 資料庫名稱

[CHARACTER SET 字元集名稱]

[COLLATE Collation 名稱]

- 只能修改資料庫字元集及**Collation**
- 修改資料庫字元集或**Collation**，不影響已經存在的表格

刪除資料庫

■ 刪除資料庫語法

DROP DATABASE [IF EXISTS] 資料庫名稱

- **IF EXISTS** 可避免資料庫已存在的錯誤
- **MySQL**執行刪除資料庫的敘述，會直接刪除
 - 不會有確認是否刪除訊息
 - 只能靠備份才可能還原資料

取得資料庫資訊

- 取得MySQL伺服器中所有資料庫的名稱

SHOW DATABASES

SHOW SCHEMAS

- 取得建立資料庫的敘述

SHOW CREATE DATABASE 資料庫名稱

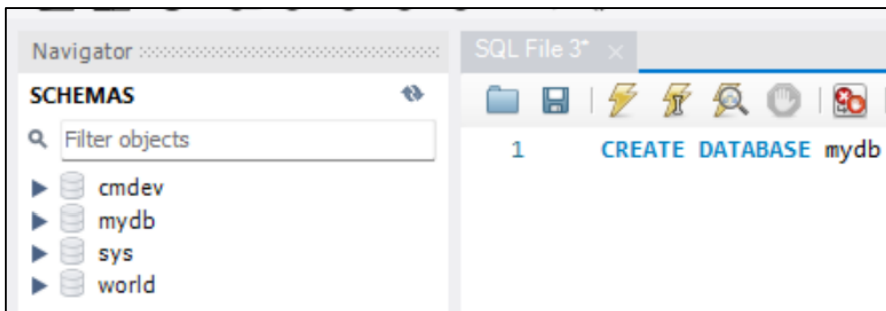
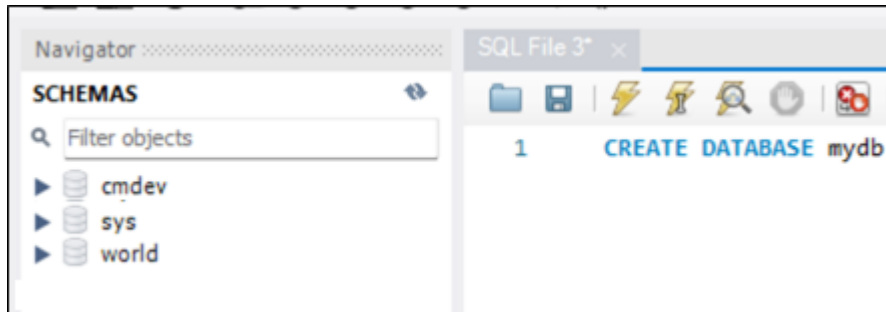
- MySQL系統資料庫 information_schema

- 資料庫相關資訊存在系統資料庫SCHEMATA表格

SELECT * FROM information_schema.SCHEMATA

範例

- 新增資料庫避免資料庫已存在錯誤訊息



✖	4	09:59:13	CREATE DATABASE mydb	Error Code: 1007. Can't create database 'mydb'; database exists
---	---	----------	----------------------	---

範例

- 新增**UTF8**字元集且不分大小寫的資料庫

範例

- 新增**big5**字元集且英文不分大小寫的資料庫

範例

- 顯示MySQL伺服器中所有資料庫

範例

- 查詢MySQL伺服器資料庫訊息
 - 使用系統資料庫 `information_schema`

範例

- 取得mydb資料庫建立的敘述

範例

- 刪除mydb資料庫

儲存引擎

■ MySQL提供三種儲存引擎

- 不同的資料儲存方式與運作特色
- MyISAM：MySQL 5.5之前預設儲存引擎
 - 簡單，運作效率比較好
 - 不支援交易 transaction
- InnoDB：MySQL 5.5版後為預設儲存引擎
 - 功能強大，接近大型商用資料庫系統
 - 交易(transaction)、回復(rollback)與鎖定(row-level locking)
- MEMORY：將資料儲存在記憶體中
 - 運作的效率最快
 - 非永續儲存，重新啟動後表格資料全部消失，只剩表格結構

儲存引擎

■ MyISAM 儲存引擎

□ 建立表格為檔名的三個檔案

- XXX.frm : 表格欄位即設定資訊
- XXX.MYD : 儲存表格記錄
- XXX.MYI : 儲存表格索引

□ 資料可攜性 **portable**

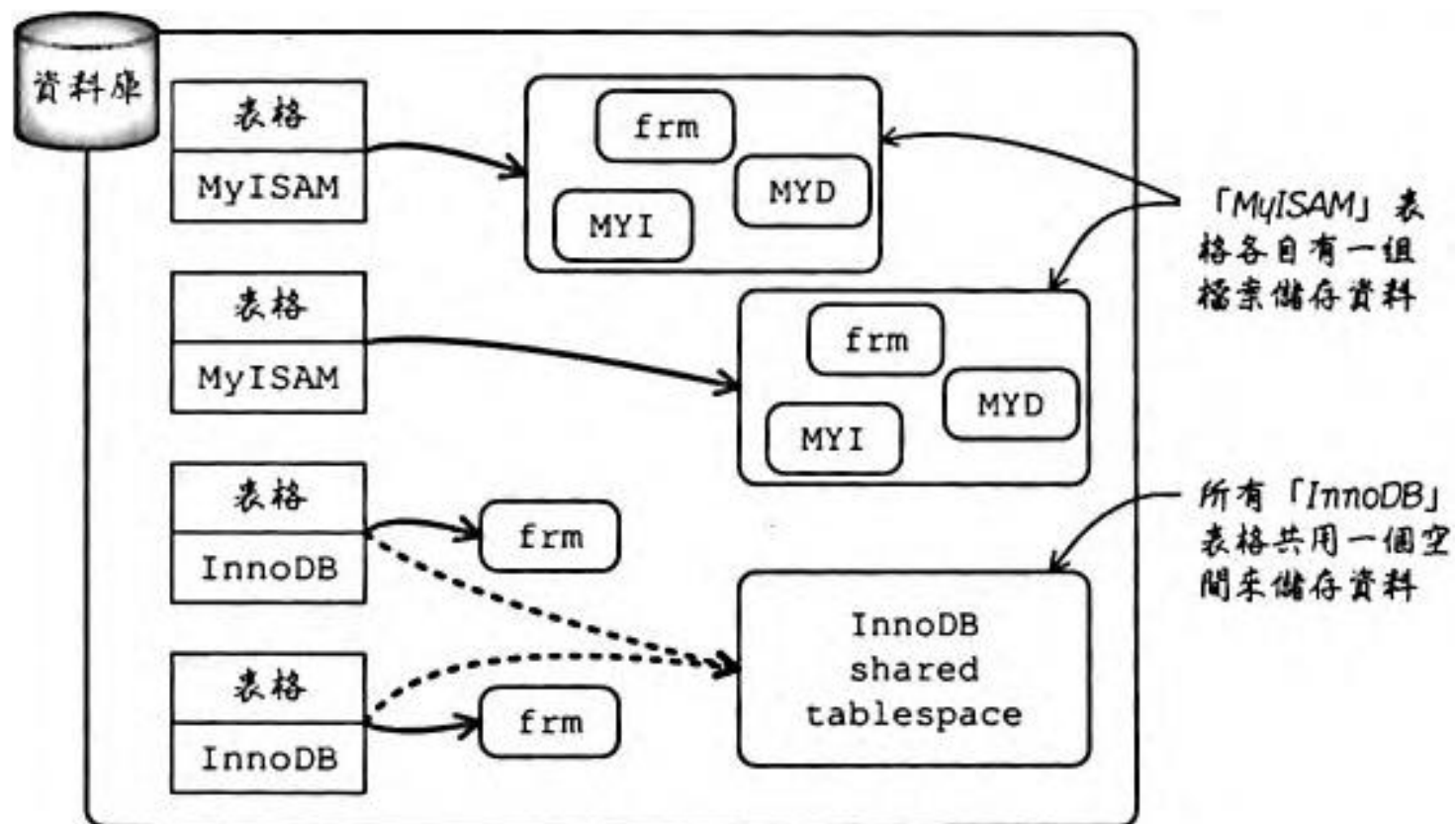
- 檔案複製即可把資料庫複製到另外一台MySQL伺服器

儲存引擎

■ InnoDB 儲存引擎

- MySQL 3.23.49 開始支援
 - 提供大型商用資料庫軟體相似的功能
 - 交易(transaction)：資料庫執行中不可分割的邏輯單位，多個資料操作(交易)一起完成或一起取消
 - 回復(rollback)：資料操作發生問題，將資料還原至交易前狀態
 - 鎖定(row-level locking)：資料庫記錄會被鎖定，才可進行讀寫
 - InnoDB資料儲存方式
 - XXX.frm：每個表格建立frm檔案，儲存表格欄位及設定資訊
 - 多個表格的記錄及索引儲存在共用檔案
-
- 共用的儲存空間中不能超過兩百萬個表格

儲存引擎儲存方式比較



交易ACID特性

■ 交易ACID特性

- ❑ 原子性（**Atomicity**）：交易作為一個整體被執行，包含在其中的對資料庫的操作不是全部被執行，就是全部都不執行
- ❑ 一致性（**Consistency**）：交易應確保資料庫的狀態從一個一致狀態轉變為另一個一致狀態
- ❑ 隔離性（**Isolation**）：多個交易並行執行時，一個交易的執行不應影響其他交易的執行
- ❑ 永續性（**Durability**）：已被提交的交易對資料庫的修改應該永久儲存在資料庫中

儲存引擎

■ MEMORY 儲存引擎

- ❑ 記錄與索引資料儲存在記憶體中
- ❑ 查詢或維護資料時的效率好
- ❑ 伺服器關閉、重新啟動或當機時，**MEMORY**儲存引擎的表格資料會全部消失，只剩下表格結構
- ❑ 不適合儲存大量資料的表格，會耗用太多記憶體
- ❑ 檔案系統中只儲存 **frm** 檔案

儲存引擎選擇

■ 儲存引擎選擇

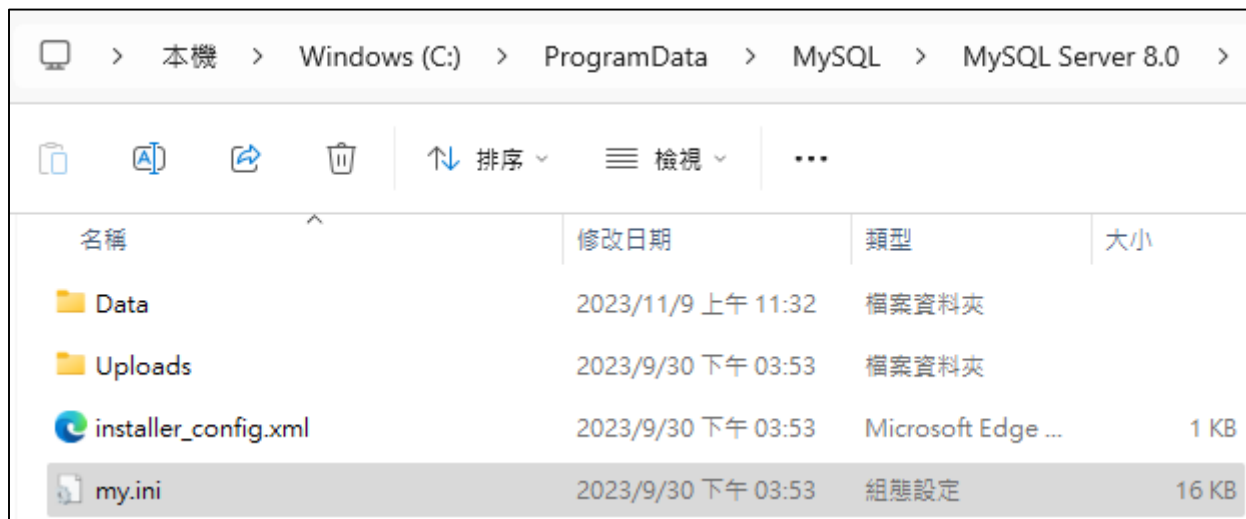
- 作業系統對檔案可能有檔案數量或檔案大小限制
- 有檔案數量限制時可選擇**InnoDB**儲存引擎
 - **InnoDB**表格共用儲存空間，使用檔案數量較少
- 有檔案大小限制時可選擇**MyISAM**儲存引擎
 - **InnoDB**共用的儲存空間中不能超過兩百萬個表格

■ 查詢支援的儲存引擎

SHOW ENGINES

資料庫伺服器設定檔 my.ini

- MySQL資料庫伺服器在啟動時讀取
 - 啟動資料庫伺服器時需要的資訊



課程大綱

- 1) 資料庫管理
- 2) 欄位資料型態
 - 數值型態 / 位元型態
 - 字串型態 / 列舉與集合型態
 - 日期與時間型態
- 3) 表格管理

數值型態

■ 整數數值資料

型態	bytes	預設長度	數字範圍	正數範圍
TINYINT[(長度)]	1	4	-128~127	0~255
SMALLINT[(長度)]	2	6	-32768~32767	0~65535
MEDIUMINT[(長度)]	3	9	$-2^{23} \sim 2^{23}-1$	$0 \sim 2^{24}-1$
INT[(長度)]	4	11	$-2^{31} \sim 2^{31}-1$	$0 \sim 2^{32}-1$
BIGINT[(長度)]	8	20	$-2^{63} \sim 2^{63}-1$	$0 \sim 2^{64}-1$

- **bytes**表示欄為儲存數值大小範圍
- 長度表示包含負號數值顯示位數
 - 不影響資料實際儲存長度
- 使用**UNSIGNED**設定只能儲存正數

數值型態

■ 浮點數數值資料

型態	bytes	預設長度	最大長度	說明
FLOAT[(長度,小數位數)]	4		255, 30	單精度浮點數
DOUBLE[(長度,小數位數)]	8		255, 30	倍精度浮點數
DECIMALUBLE NUMERIC [(長度 [,小數位數])]		10, 0	65, 30	自行指定位數的 精確度

□ FLOAT與DOUBLE 儲存數值近似值

- 儲存空間小，執行運算快
- 查詢或運算可能有誤差

□ DECIMAL 或 NUMERIC 儲存完全精準的數值

- 佔用較大儲存空間
- 查詢或運算不會有誤差

數值型態

■ 浮點數數值資料

- 小數位數表示小數點後顯示位數
 - 小數位數超過部分會四捨五入存入
 - 長度表示數值顯示位數
 - 包含負號不包含小數點
 - 資料不可以超過最大長度
 - 小數位數不可以超過長度
 - 長度與小數位數一樣的時候，表示只可以儲存小數
 - 使用**UNSIGNED**設定只能儲存正數
-
- **ZEROFILL**設定回傳的資料左側根據長度填充0

位元數值型態

■ 以二進位的型式儲存數值資料


型態	預設長度	最大長度	數值範圍
BIT[(長度)]	1	64	$0 \sim 2^{\text{長度}} - 1$

- BIT：0或1，可用來儲存布林值
- BIT(8)：0~255
- SQL語法中用 b'01010011' 表示2進位數值

範例

- 下列命令執行結果為何？

Query 1 cmdev.integertable x			
Info Columns Indexes Triggers Foreign keys Partitions			
Column	Type ▲	Default Value	Nullable
◇ n5	bigint		YES
◇ n4	int		YES
◇ n3	mediumint		YES
◇ n2	smallint		YES
◇ n	tinyint		YES

Query 1 x integertable	
	
1	INSERT INTO integertable VALUES (1, 2, 3, 4, 5);
2	INSERT INTO integertable VALUES (1.2, 2.3, 3.4, 4.5, 5.6);
3	INSERT INTO integertable VALUES (1, 65536, 3, 4, 5);
4	

範例

- 下列命令執行結果為何？

Query 1 cmdev.numerictable x			
Info Columns Indexes Triggers Foreign keys Partitions Gr			
Column	Type	Default Value	Nullable
◇ f	float unsigned		YES
◇ f2	double		YES
◇ f3	decimal(10,0) unsig...		YES
◇ i	tinyint unsigned		YES
◇ i2	smallint unsigned		YES
◇ i3	mediumint unsigned		YES
◇ i4	int		YES
◇ i5	bigint unsigned		YES

```
Query 1 x numerictable
1 • INSERT INTO numerictable (i4, f2) VALUES (1, 1), (-1, -1);
2 • INSERT INTO numerictable (i, f) VALUES (-1, -1);
3
```

範例

- 下列命令執行結果為何？

Query 1 cmdev.numerictable			
Info Columns Indexes Triggers Foreign keys Partitions Gr			
Column	Type	Default Value	Nullable
◇ f	float unsigned		YES
◇ f2	double		YES
◇ f3	decimal(10,0) unsig...		YES
◇ i	tinyint unsigned		YES
◇ i2	smallint unsigned		YES
◇ i3	mediumint unsigned		YES
◇ i4	int		YES
◇ i5	bigint unsigned		YES

```
Query 1 x numerictable
Limit to 1000 rows
1 • INSERT INTO numerictable (i, i2, i3, i4, i5)
2     VALUES (1, 1, 1, 1, 1),
3             (123, 123, 123, 123, 123),
4             (123, 12345, 1234567, 1234567890, 1234567890123456789);
5
6 • INSERT INTO numerictable (f, f2, f3)
7     VALUES (123.12, 123.12, 123.12),
8             (123.123, 123.123, 123.123);
9
10 • INSERT INTO numerictable (f, f2, f3) VALUES (0.1, 12345.12, 0.1);
11
```

範例

- 下列命令執行結果為何？

Query 1				cmdev.numerictable3			
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants	DDL
Column	Type	Default Value	Nullable				
◇ f	float(5,2) unsigned zerofill		YES				
◇ f2	double(7,3) unsigned zerofill		YES				
◇ f3	decimal(9,5) unsigned zerofill		YES				
◇ i	tinyint(3) unsigned zerofill		YES				
◇ i2	smallint(4) unsigned zerofill		YES				
◇ i3	mediumint(5) unsigned zerofill		YES				
◇ i4	int(6) unsigned zerofill		YES				
◇ i5	bigint(7) unsigned zerofill		YES				

```
Query 1 x cmdev.numerictable3
Limit to 1000 rows
1 • INSERT INTO numerictable3 VALUES (1, 1, 1, 1, 1, 1.1, 1.1, 1.1);
2
3 • INSERT INTO numerictable3 (i4) VALUES (123), (1234567);
4
```

範例

- 下列命令執行結果為何？

Query 1 cmdev.bittable x			
Info Columns Indexes Triggers Foreign keys Partit			
Column	Type	Default Value	Nullable
◇ n	bit(1)		YES
◇ n2	bit(8)		YES
◇ n3	bit(64)		YES

Query 1 x cmdev.bittable	
Limit to 1000 rows	
1 •	INSERT INTO bittable
2	VALUES (1, 255, 65536),
3	(b'1', b'1111111', b'111111111111111')
4	

字串型態

- 字串分為非二進位制與二進位制字串
- 非二進位制字串型態
 - 儲存一般文字的字串，會有特定的字元集與collation

型態	最大長度	實際儲存空間	說明
CHAR[(長度)]	255	指定長度	固定長度的字串
VARCHAR[(長度)]	65535	字元個數加1或2bytes	變動長度的字串
TINYTEXT	255	字元個數加1bytes	
TEXT	65535	字元個數加2bytes	
MEDIUMTEXT	$2^{24}-1$	字元個數加3bytes	
LONGTEXT	$2^{32}-1$	字元個數加4bytes	

字元集與collation

- 不同字元集會佔用不同的儲存空間
 - Latin1 1 byte ; big5 2 byte ; UTF-8 3 byte ;
 - LENGTH()函式傳回字串實際儲存長度
 - CHAR_LENGTH()函式傳回字串的字元數量
- 不同collation會影響字串排列順序及查詢結果
 - XXX_cs : 字串區分大小寫
 - XXX_ci : 字串不區分大小寫

字串型態

■ 二進位制字串型態

- 使用位元組儲存資料，不包含字元集與collation
- 大多用來儲存圖片或音效

型態	最大長度	實際儲存空間	說明
BINARY[(長度)]	255	指定長度	固定長度的字串
VARBINARY[(長度)]	65535	字元個數加1或2bytes	變動長度的字串
TINYBLOB	255	字元個數加1bytes	
BLOB	65535	字元個數加2bytes	
MEDIUMBLOB	$2^{24}-1$	字元個數加3bytes	
LOB	$2^{32}-1$	字元個數加4bytes	

列舉字串型態

- 特殊的非二進位制字串型態

- 列舉 **ENUM**

ENUM(字串值[,...])

- 宣告欄位時將型態宣告為**ENUM**
 - 宣告有限數量列舉字串
 - **MySQL**會為列舉型態字串從1開始編號
 - 真正儲存在資料庫中的資料是成員編號
 - 新增或修改資料時，欄位可使用一個成員字串或編號
 - 檢查輸入值的正確性

集合字串型態

- 特殊的非二進位制字串型態

- 集合 SET

- SET(字串值[,...])

- 宣告欄位時將型態宣告為SET
 - 宣告有限數量列舉字串
 - MySQL會為集合型態字串以2的次方數開始編號(1, 2, 4, 8..)
 - 真正儲存在資料庫中的資料是成員編號
 - 新增或修改資料時，欄位可儲存多個成員字串或編號
 - 可以將編號集合加總為一個數字來設定
 - 檢查輸入值的正確性

範例

- 下列命令執行結果為何？

Query 1 cmdev.nonbinarytable x					
Info Columns Indexes Triggers Foreign keys Partitions Grants DDL					
Column	Type	Default Value	Nullable	Character Set	Collation
s	char(10)		YES	big5	big5_chinese_ci
s2	varchar(10)		YES	big5	big5_chinese_ci

```
Query 1 x cmdev.nonbinarytable
Limit to 1000 rows
1 • INSERT INTO nonbinarytable
2 VALUES ('', ''), ('123', '123'), ('1234567890', '1234567890')
3
```

範例

■ 下列命令執行有什麼不同？

Query 1		nonbinarytable2		cmdev.nonbinarytable2		x		
Info		Columns	Indexes	Triggers	Foreign keys	Partitions	Grants	DDL
Column		Type	Default Value		Nullable	Character Set	Collation	
◆ s		varchar(6)			YES	latin1	latin1_swedish_ci	
◆ s2		varchar(6)			YES	big5	big5_chinese_ci	
◆ s3		varchar(6)			YES	utf8mb3	utf8_general_ci	

	s	s2	s3
▶	abc	abc	abc
	abcdef	abcdef	abcdef
	abc	一二三	一二三
	abcdef	一二三四五六	一二三四五六
	abcdef	一二三abc	一二三abc

Query 1 x nonbinarytable2 cmdev.nonbinary	
1 • SELECT s, LENGTH(s),	
2 s2, LENGTH(s2),	
3 s3, LENGTH(s3)	
4 FROM nonbinarytable2;	
5	

Query 1 x nonbinarytable2 cmdev.nonbinary	
6 • SELECT s, CHAR_LENGTH(s),	
7 s2, CHAR_LENGTH(s2),	
8 s3, CHAR_LENGTH(s3)	
9 FROM nonbinarytable2;	
10	

範例

■ 下列命令執行結果為何？

Query 1 cmdev.nonbinarytable3 x						
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants DDL
Column	Type	Default Value	Nullable	Character Set	Collation	
◆ n	int		YES			
◆ s	varchar(20)		YES	latin1	latin1_general_ci	
◆ s2	varchar(20)		YES	latin1	latin1_general_cs	

	n	s	s2
▶ 1	aaa	aaa	
2	AAA	AAA	
3	bbb	bbb	
4	BBB	BBB	
5	abc	abc	
6	ABC	ABC	

Query 1 x cmdev.nonbinarytable3 nonbinarytable3	
1	• SELECT * FROM nonbinarytable3;
2	
3	• SELECT *
4	FROM nonbinarytable3
5	ORDER BY s;
6	
7	• SELECT *
8	FROM nonbinarytable3
9	ORDER BY s2;
10	
11	• SELECT *
12	FROM nonbinarytable3
13	WHERE s = 'aaa';
14	
15	• SELECT *
16	FROM nonbinarytable3
17	WHERE s2 = 'aaa';

範例

Query 1			
cmdev.binarytable × binarytable			
Info	Columns	Indexes	Triggers
Columns	Indexes	Triggers	Foreign keys
Columns	Indexes	Triggers	Partitions
Column	Type	Default Value	Nullable
◇ b	binary(6)		YES
◇ b2	varbinary(6)		YES

	b	b2
▶	BLOB	BLOB
	BLOB	BLOB
	BLOB	BLOB

範例

Query 1 cmdev.enumtable x enumtable						
Info Columns Indexes Triggers Foreign keys Partitions Grants DDL						
Column	Type	Default Value	Nullable	Character Set	Collation	
enumsize	enum('XS','S','M','L','XL')		YES	big5	big5_chinese_ci	
stringsize	varchar(2)		YES	big5	big5_chinese_ci	

■ 下列命令執行結果為何？

```
Query 1 x cmdev.enumtable enumtable
Limit to 1000 rows
1 • INSERT INTO enumtable
2   VALUES ('XS', 'XS'),('S', 'S'),('M', 'M'),
3     ('L', 'L'),('XL', 'XL');
4
5 • INSERT INTO enumtable (stringsize) VALUES ('QQ');
6
7 • INSERT INTO enumtable (enumsize) VALUES ('QQ');
8
9 • INSERT INTO enumtable (enumsize)
10  VALUES ('XS'),('S'),('M'),('L'),('XL');
11
12 • INSERT INTO enumtable (enumsize)
13  VALUES (1),(2),(3),(4),(5);
```

範例

- 下列命令執行結果為何？

	enumsize	stringsize
▶	XS	XS
	S	S
	M	M
	L	L
	XL	XL
	NULL	QQ
	XS	NULL
	S	NULL
	M	NULL
	L	NULL
	XL	NULL
	XS	NULL
	S	NULL
	M	NULL
	L	NULL
	XL	NULL

```
Query 1 x cmdev.enumtable enumtable
1 • SELECT enumsize
2 FROM enumtable
3 ORDER BY enumsize;
4
5 • SELECT stringsize
6 FROM enumtable
7 ORDER BY stringsize;
8
9 • SELECT * FROM enumtable
10 WHERE enumsize = 'M';
11
12 • SELECT * FROM enumtable
13 WHERE enumsize = 3;
14
```

範例

Query 1 cmdev.settable x					
Info Columns Indexes Triggers Foreign keys Partitions Grants DDL					
Column	Type	Default ...	Nullable	Character Set	Collation
workingday	set('MON','TUE','WED','THU','FRI','SAT','SUN')		YES	big5	big5_chinese_ci

■ 下列命令執行結果為何？

```
Query 1 x cmdev.settable settable
Limit to 1000 rows
1 • INSERT INTO settable
2   VALUES ('MON,WED,FRI'),
3           ('TUE,THU'),
4           ('SAT,SUN'),
5           ('MON,TUE,WED,THU,FRI,SAT,SUN');
6
7 • INSERT INTO settable VALUES ('MON,HELLO,FRI');
8
9 • INSERT INTO settable VALUES (0),(1),(4),(16);
10
11 • INSERT INTO settable VALUES (''),('MON'),('WED'),('FRI');
12
13 • INSERT INTO settable VALUES ('MON,WED,FRI');
14
15 • INSERT INTO settable VALUES (21);
```


範例

■ 下列命令執行結果為何？

Query 1 cmdev.establish x					
Info Columns Indexes Triggers Foreign keys Partitions Grants DDL					
Column	Type	Default...	Nullable	Character Set	Collation
◇ enumsize	enum('XS','S','M','L','XL')		YES	latin1	latin1_general_ci
◇ enumsize2	enum('XS','S','M','L','XL')		YES	latin1	latin1_general_cs
◇ workingday	set('MON','TUE','WED','THU','FRI','SAT','SUN')		YES	latin1	latin1_general_ci
◇ workingday2	set('MON','TUE','WED','THU','FRI','SAT','SUN')		YES	latin1	latin1_general_cs

```
Query 1 x cmdev.establish estable
1 • INSERT INTO estable
2   VALUES (1, 1, 21, 21),
3           ('M', 'M', 'MON', 'MON'),
4           ('m', 'M', 'mon', 'MON');
5
6 • INSERT INTO estable (enumsize2, workingday2)
7   VALUES ('m', 'mon');
```

日期與時間型態

■ 日期與時間型態

型態	bytes	說明	範圍
DATE	3	日期	'1000-01-01' ~ '9999-12-31'
TIME	3	時間	'-838:59:59' ~ '838:59:59'
YEAR[(4 2)]	1	西元年	1901~2155(YYYY) 1970~2069(YY)
DATETIME	8	日期與時間	'1000-01-01 00:00:00' ~ '9999-12-31 23:59:59'
TIMESTAMP	4	日期與時間	'1970-01-01 00:00:00' ~ '2037....'

日期與時間型態

■ DATE型態

- 年份可使用四個或兩個數字表示西元年
 - 兩個數字：(19)70~(19)99, (20)00~(20)69
 - 年份非兩個數字：就以四位西元年解讀，前面0可省略

■ TIME型態

- 指的是時間間隔而不是特定時點
- 秒或分可被省略，省略時都設定為0

日期與時間型態

■ YEAR型態

- 只需要儲存年份資料，節省空間
 - 四個數字：四位西元年略
 - 兩個數字：(19)70~(19)99, (20)00~(20)69
 - 一個數字：200X年

■ DATETIME型態

- 儲存完整的年、月、日與時、分、秒資料
- 日期與時間需有一個以上空白
- 時間部份為 00:00:00~23:59:59
- 時、分、秒都可以省略，省略時都設定為0

日期與時間型態

- **TIMESTAMP**型態
 - 格式與**DATETIME**一樣
 - 儲存空間只有4個bytes
 - 具有時區特性
 - MySQL資料庫預設使用作業系統時區設定
- 查詢系統及用戶時區設定

```
SELECT @@GLOBAL.TIME_ZONE
SELECT @@SESSION.TIME_ZONE
```

設定系統及用戶時區

- 設定系統及用戶時區
 - 以格林威治標準時區為基準
 - ± 時時:分分

SET SESSION TIME_ZONE = '-08:00'

SET GLOBAL TIME_ZONE = '+08:00'

範例

- 下列命令執行結果為何？

Query 1 cmdev.dttable x				
Info	Columns	Indexes	Triggers	Foreign keys
Partition				
Column	Type	Default Value ▲		Nullable
◇ d	date			YES
◇ dt	datetime			YES
◇ t	time			YES
◇ ts	timestamp			YES
◇ y4	year			YES

```
Query 1 x cmdev.dttable dttable
Limit to 1000 rows
1 • INSERT INTO dttable (d) VALUES ('9000-1-1');
2
3 • INSERT INTO dttable (d) VALUES ('900-1-1');
4
5 • INSERT INTO dttable (d) VALUES ('90-1-1');
6
7 • INSERT INTO dttable (d) VALUES ('9-1-1');
8
9 • INSERT INTO dttable (d) VALUES ('2000-1-1');
10
11 • INSERT INTO dttable (d) VALUES ('200-1-1');
12
13 • INSERT INTO dttable (d) VALUES ('20-1-1');
14
15 • INSERT INTO dttable (d) VALUES ('2-1-1');
```

範例

- 下列命令執行結果為何？

Query 1 cmdev.dttable x			
Info Columns Indexes Triggers Foreign keys Partition			
Column	Type	Default Value ▲	Nullable
◇ d	date		YES
◇ dt	datetime		YES
◇ t	time		YES
◇ ts	timestamp		YES
◇ y4	year		YES

Query 1 x cmdev.dttable dttable	
Limit to 1000 rows	
1 •	INSERT INTO dttable (t) VALUES ('200:30:00');
2	
3 •	INSERT INTO dttable (t) VALUES ('-1:20:30');
4	
5 •	INSERT INTO dttable (t) VALUES ('200:30:30');
6	
7 •	INSERT INTO dttable (t) VALUES ('200:30');
8	
9 •	INSERT INTO dttable (t) VALUES ('200');

範例

- 下列命令執行結果為何？

Query 1 cmdev.dttable x			
Info Columns Indexes Triggers Foreign keys Partition			
Column	Type	Default Value ▲	Nullable
◇ d	date		YES
◇ dt	datetime		YES
◇ t	time		YES
◇ ts	timestamp		YES
◇ y4	year		YES

Query 1 x cmdev.dttable dttable	
1 • INSERT INTO dttable (dt)	
2 VALUES ('2000-01-01 10:10:10');	
3	
4 • INSERT INTO dttable (dt)	
5 VALUES ('2000-01-01 10:10:10');	
6	
7 • INSERT INTO dttable (dt)	
8 VALUES ('2000-01-01');	
9	
10 • INSERT INTO dttable (dt)	
11 VALUES ('2000-01-01 200:00:00');	

範例

■ 下列命令執行結果為何？

Query 1 cmdev.dtttable				
Info	Columns	Indexes	Triggers	Foreign keys
Partition				
Column	Type	Default Value	Nullable	
d	date		YES	
dt	datetime		YES	
t	time		YES	
ts	timestamp		YES	
y4	year		YES	

```
Query 1 x cmdev.dtttable dtttable
Limit to 1000 rows
1 • INSERT INTO dtttable (y4) VALUES ('2000'),(2000);
2
3 • INSERT INTO dtttable (y4) VALUES ('200'),(200);
4
5 • INSERT INTO dtttable (y4) VALUES ('20'),(20);
6
7 • INSERT INTO dtttable (y4) VALUES ('2'),(2);
8
9 • INSERT INTO dtttable (y4) VALUES ('0000'), (0);
10
11 • INSERT INTO dtttable (y4) VALUES ('000');
12
13 • INSERT INTO dtttable (y4) VALUES ('00');
14
15 • INSERT INTO dtttable (y4) VALUES ('0');
```

範例

- 下列命令執行結果為何？

Query 1 cmdev.dttable x			
Info Columns Indexes Triggers Foreign keys Partition			
Column	Type	Default Value ▲	Nullable
◇ d	date		YES
◇ dt	datetime		YES
◇ t	time		YES
◇ ts	timestamp		YES
◇ y4	year		YES

Query 1 x cmdev.dttable dttable		
Limit to 1000 rows		
1	•	INSERT INTO dttable (ts) VALUES ('1970-1-01 8:0:1');
2		
3	•	INSERT INTO dttable (ts) VALUES ('1970-1-01 0:0:1');
4		

範例

- 下列命令執行結果為何？

Query 1 cmdev.dttable			
Info Columns Indexes Triggers Foreign keys Partition			
Column	Type	Default Value ▲	Nullable
◇ d	date		YES
◇ dt	datetime		YES
◇ t	time		YES
◇ ts	timestamp		YES
◇ y4	year		YES

```
Query 1 x cmdev.dttable dttable
Limit to 1000 rows
1 • SET SESSION TIME_ZONE = '+00:00';
2 • SET GLOBAL TIME_ZONE = '+00:00';
3 • SELECT @@GLOBAL.TIME_ZONE, @@SESSION.TIME_ZONE;
4
5 • INSERT INTO dttable (dt, ts)
6   VALUES ('2000-01-01 00:00:00', '2000-01-01 00:00:00');
7
8 • SELECT dt,ts FROM dttable;
9
10 • SET SESSION TIME_ZONE = '-08:00';
11 • SELECT dt,ts FROM dttable;
12
```

課程大綱

- 1) 資料庫管理
- 2) 欄位資料型態
- 3) 表格管理
 - 建立表格
 - 修改與刪除表格
 - 查詢表格資訊

建立表格

■ 建立表格

CREATE TABLE [IF NOT EXISTS] 表格名稱

(欄位名稱 欄位型態 [欄位屬性],

...)

[{ENGINE | TYPE} [=] 儲存引擎名稱]

[CHARACTER SET [=] 字元集名稱]

[COLLATE [=] collation名稱]

□ IF NOT EXISTS，避免表格已經存在發生錯誤

□ CHARACTER SET 可縮寫為CHARSET

□ = 可省略

通用欄位屬性

■ 通用欄位屬性

- 所有型態的欄位屬性
- NOT NULL / NULL：不能/可以儲存NULL值
- DEFAULT [預設值]：未指定值時使用之預設值
 - 預設值需符合資料型態及NULL狀態設定
 - BLOB與TEXT欄位不可以使用預設值
 - 如未設定預設值，預設值為NULL
- [UNIQUE [KEY]：欄位為不可重複之索引
- [PRIMARY] KEY]：欄位為主鍵值，不可重複且不可為NULL

字串欄位屬性

- 字串型態欄位
 - CHAR(長度)
 - VARCHAR(長度)
 - TINYTEXT
 - TEXT
 - MEDIUMTEXT
 - LONGTEXT
 - ENUM(字串值[,...])
 - SET(字串值[,...])
- 可設定 **CHARACTER SET** 及 **COLLATE** 屬性

數值欄位屬性

■ 數值欄位屬性

- TINYINT[(長度)]
- SMALLINT[(長度)]
- MEDIUMINT[(長度)]
- INT[(長度)]
- INTEGER[(長度)]
- BIGINT[(長度)]
- REAL[(長度,小數位數)]
- DOUBLE[(長度,小數位數)]
- FLOAT[(長度,小數位數)]
- DECIMAL[(長度,小數位數)]
- NUMERIC[(長度,小數位數)]

數值欄位屬性

- 可設定屬性
 - UNSIGNED：不能儲存負數
 - ZEROFILL：依長度填充0
 - AUTO_INCREMENT：自動遞增，與索引有關

範例

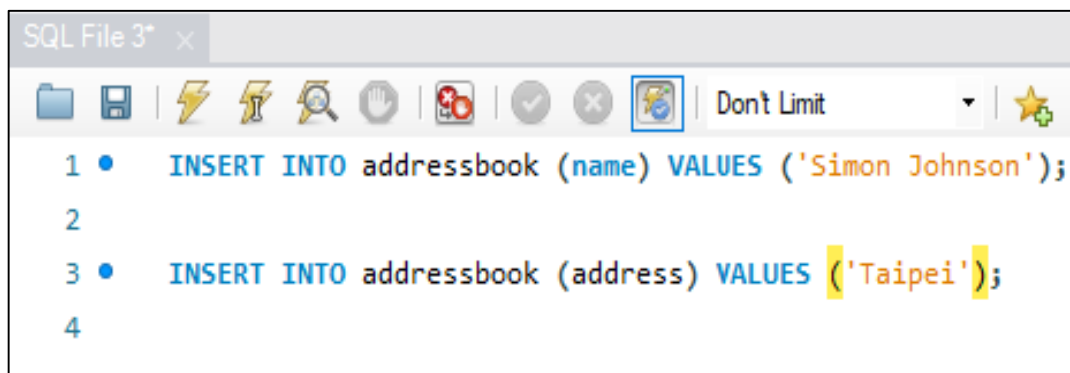
- 在mydb資料庫中新增addressbook資料表
 - Name varchar
 - Tel varchar
 - Address varchar
 - Birthdate Date
 - 避免重複建立
 - InnoDB
 - Utf8
 - Utf8_ci

範例

- 在mydb資料庫中新增addressbook資料表
 - Name varchar 不可為NULL
 - Tel varchar 可為NULL
 - Address varchar
 - Birthdate Date
 - 檢視建立敘述

範例

- 依據上題表格限制，下列命令執行結果為何？



```
SQL File 3* x
1 • INSERT INTO addressbook (name) VALUES ('Simon Johnson');
2
3 • INSERT INTO addressbook (address) VALUES ('Taipei');
4
```

範例

- 重新建立 addressbook 資料表
 - 地址預設值設為 Taipei
 - 檢視新增 'Simon Johnson' 結果

TIMESTAMP欄位屬性

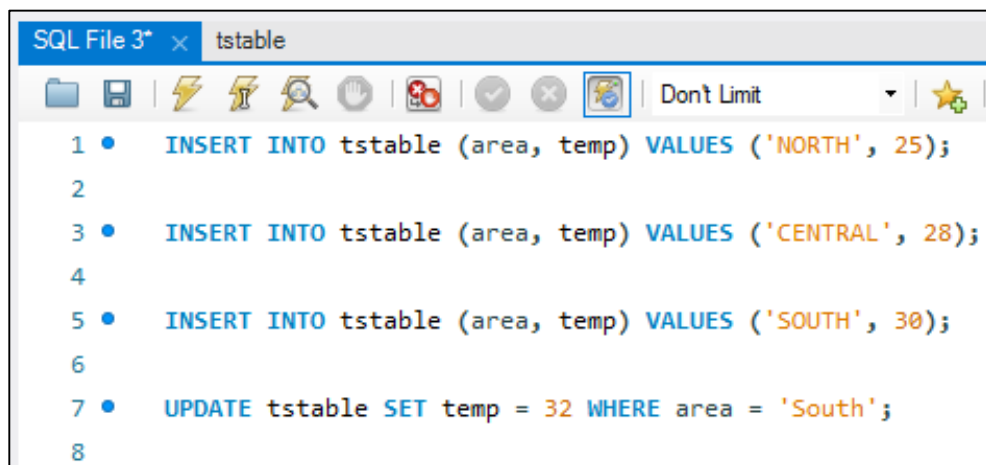
- 日期時間資料
- 預設屬性
 - NOT NULL
 - DEFAULT：預設資料
 - ON UPDATE：新增修改時自動填入當前日期時間資料
 - 'YYYY-MM-DD HH:MM:SS' 指定特定日期時間
 - CURRENT_TIMESTAMP 表示填入目前的日期時間
 - 表格中只能有一個TIMESTAMP欄位使用CURRENT_TIMESTAMP
 - 0 新增修改時若該欄位值為NULL，自動填入當前日期時間
 - 第一個TIMESTAMP欄位預設為ON UPDATE，後續則否

範例

- 建立表格tstable
 - ts : TimeStamp
 - ts2 : TimeStamp
 - area : 不可為NULL字串
 - temp : 不可為NULL整數
- 檢視建立敘述

範例

- 依據上題表格限制，下列命令執行結果為何？



```
SQL File 3* x tstable
1 • INSERT INTO tstable (area, temp) VALUES ('NORTH', 25);
2
3 • INSERT INTO tstable (area, temp) VALUES ('CENTRAL', 28);
4
5 • INSERT INTO tstable (area, temp) VALUES ('SOUTH', 30);
6
7 • UPDATE tstable SET temp = 32 WHERE area = 'South';
8
```

範例

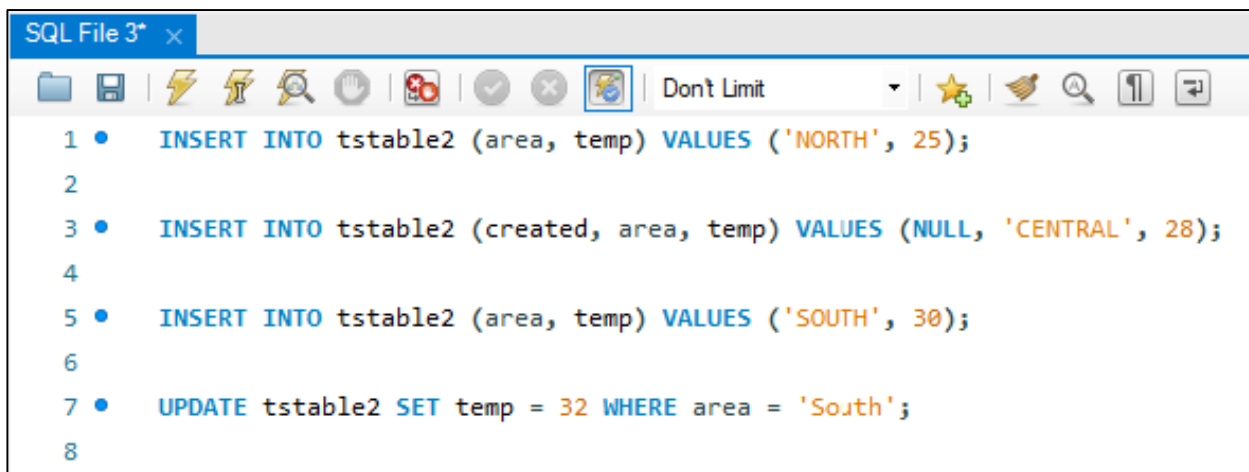
■ 建立表格tstable2

- created : TimeStamp 預設為資料新增時間
- updated : TimeStamp 預設為NULL, 記錄資料修改時間
- area : 不可為NULL字串
- temp : 不可為NULL整數

■ 檢視建立敘述

範例

- 依據上題表格限制，下列命令執行結果為何？



```
SQL File 3* x
1 • INSERT INTO tstable2 (area, temp) VALUES ('NORTH', 25);
2
3 • INSERT INTO tstable2 (created, area, temp) VALUES (NULL, 'CENTRAL', 28);
4
5 • INSERT INTO tstable2 (area, temp) VALUES ('SOUTH', 30);
6
7 • UPDATE tstable2 SET temp = 32 WHERE area = 'South';
8
```

使用現有表格建立新表格

- 使用一個現有的表格來建立新的表格
 - 以查詢傳回的結果來建立

- 語法

CREATE TABLE [IF NOT EXISTS] 表格名稱

[(欄位定義，...)]

[表格屬性]

查詢敘述

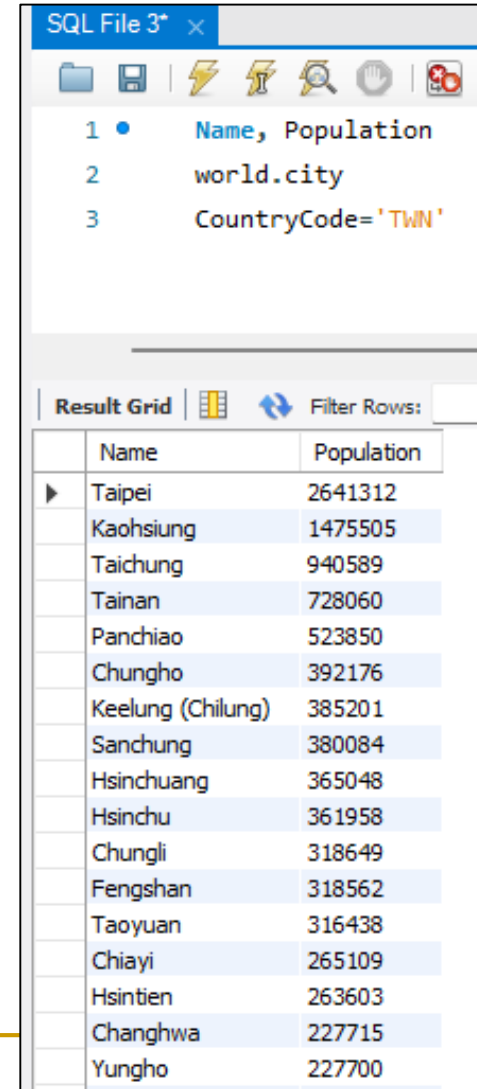
暫存表格

- 建立特定用戶暫存表格
 - 表格專屬於特定用戶，用戶離線後，表格會被自動刪除
 - 其它用戶不能使用，不同用戶可使用重複名稱
 - 與資料庫中的表格名稱相同，原資料庫表格會被遮蔽
 - 語法

CREATE TEMPORARY TABLE [IF NOT EXISTS] 表格名稱

範例

- 由 world.city 表格建立 cityofTaiwan 表格
 - CountryCode為TWN
 - 使用原表格Name, Population欄位名稱與定義



The screenshot shows a SQL query window titled "SQL File 3*" with a toolbar at the top. The query text is as follows:

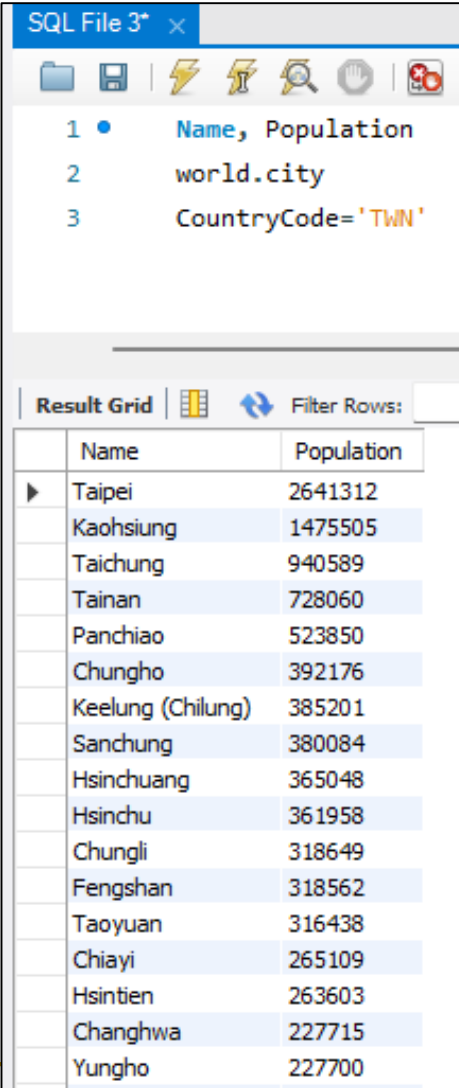
```
1 • Name, Population
2 world.city
3 CountryCode='TWN'
```

Below the query, the "Result Grid" tab is active, displaying a table of results. The table has two columns: "Name" and "Population". The results list various cities in Taiwan along with their populations.

Name	Population
Taipei	2641312
Kaohsiung	1475505
Taichung	940589
Tainan	728060
Panchiao	523850
Chungho	392176
Keelung (Chilung)	385201
Sanchung	380084
Hsinchuang	365048
Hsinchu	361958
Chungli	318649
Fengshan	318562
Taoyuan	316438
Chiayi	265109
Hsintien	263603
Changhwa	227715
Yungho	227700

範例

- 由 world.city 表格建立 cityofTaiwan2 表格
 - CountryCode為TWN
 - Name：大小為30 字串
 - Population：非負整數



The screenshot shows a SQL IDE window titled 'SQL File 3*'. The query editor contains the following SQL statement:

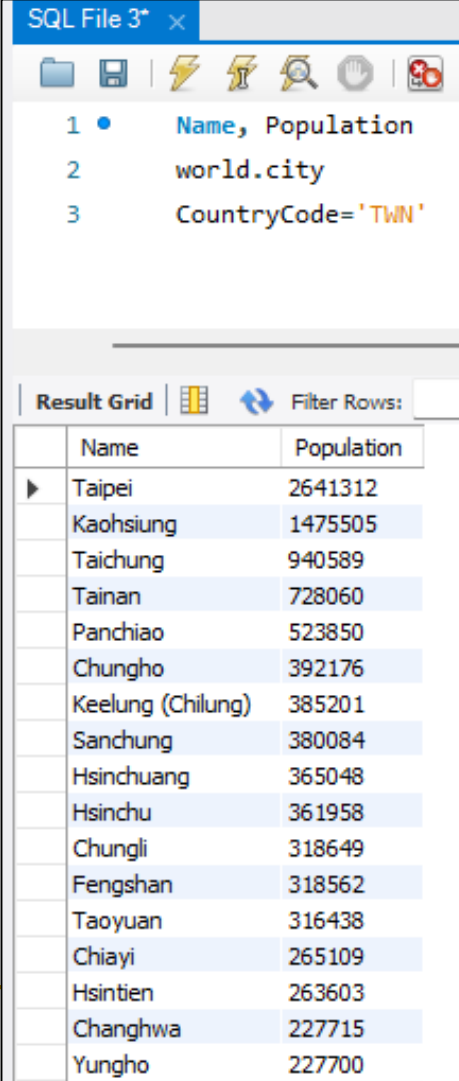
```
1 • Name, Population
2 world.city
3 CountryCode='TWN'
```

Below the query editor, the 'Result Grid' tab is active, displaying the results of the query. The results are shown in a table with two columns: 'Name' and 'Population'. The table lists 17 cities from Taipei to Yunggho, with their respective populations.

Name	Population
Taipei	2641312
Kaohsiung	1475505
Taichung	940589
Tainan	728060
Panchiao	523850
Chungho	392176
Keelung (Chilung)	385201
Sanchung	380084
Hsinchuang	365048
Hsinchu	361958
Chungli	318649
Fengshan	318562
Taoyuan	316438
Chiayi	265109
Hsintien	263603
Changhwa	227715
Yunggho	227700

範例

- 由 world.city 表格建立 cityofTaiwan3 表格
 - world.city表格中CountryCode為TWN
 - Name：大小為 30 字串
 - Population：非負整數
 - Description：大小為 50 字串



The screenshot shows a SQL File 3* window with a query editor and a result grid. The query editor contains the following SQL statement:

```
1 Name, Population
2 world.city
3 CountryCode='TWN'
```

The result grid displays the following data:

	Name	Population
▶	Taipei	2641312
	Kaohsiung	1475505
	Taichung	940589
	Tainan	728060
	Panchiao	523850
	Chungho	392176
	Keelung (Chilung)	385201
	Sanchung	380084
	Hsinchuang	365048
	Hsinchu	361958
	Chungli	318649
	Fengshan	318562
	Taoyuan	316438
	Chiayi	265109
	Hsintien	263603
	Changhwa	227715
	Yungho	227700

修改表格

■ 修改表格

- 修改表格的結構 Schema

- 語法

ALTER TABLE 表格名稱 修改定義 [...]

- 增加欄位

- 修改欄位

- 刪除欄位

- 修改表格名稱

修改表格

■ 增加表格欄位

□ 語法

ADD [COLUMN] 欄位定義 [FIRST | AFTER 欄位名稱]

- **FIRST**：新增的欄位放在第一個
- **AFTER**：新增的欄位放在指定欄位之後
- 未指定欄位位置：新增在最後一個欄位

ADD [COLUMN] (欄位定義 [,...])

- 增加多個新欄位在尾端

修改表格

- 修改表格欄位名稱、型態、大小範圍或其它屬性

- 語法

CHANGE [COLUMN] 原欄位 新欄位定義
[FIRST|AFTER 欄位]

- 可修改欄位名稱、定義及位置

- **FIRST**：修改第一個欄位名稱
 - **AFTER**：修改指定欄位之後的欄位名稱

MODIFY [COLUMN] 欄位定義 [FIRST|AFTER 欄位]

- 可修改指定欄位的定義及位置

修改表格

- 刪除欄位

- 語法

DROP [COLUMN] 欄位名稱

- 修改表格名稱

- 語法

ALTER TABLE 舊表格名稱 RENAME [TO] 新表格名稱

RENAME TABLE 舊表格名稱 TO 新表格名稱

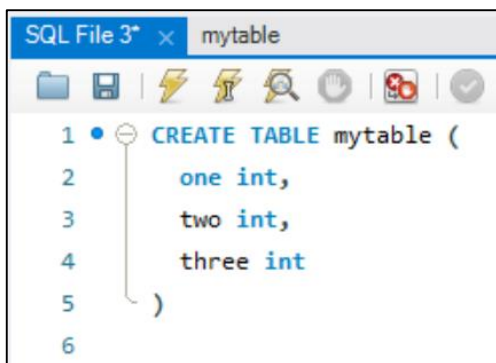
刪除表格

■ 刪除表格

- ❑ **DROP TABLE [IF EXISTS]** 表格名稱 [...]
- ❑ **IF EXISTS** 避免表格不存在發生錯誤
- ❑ 不會二次確認直接刪除

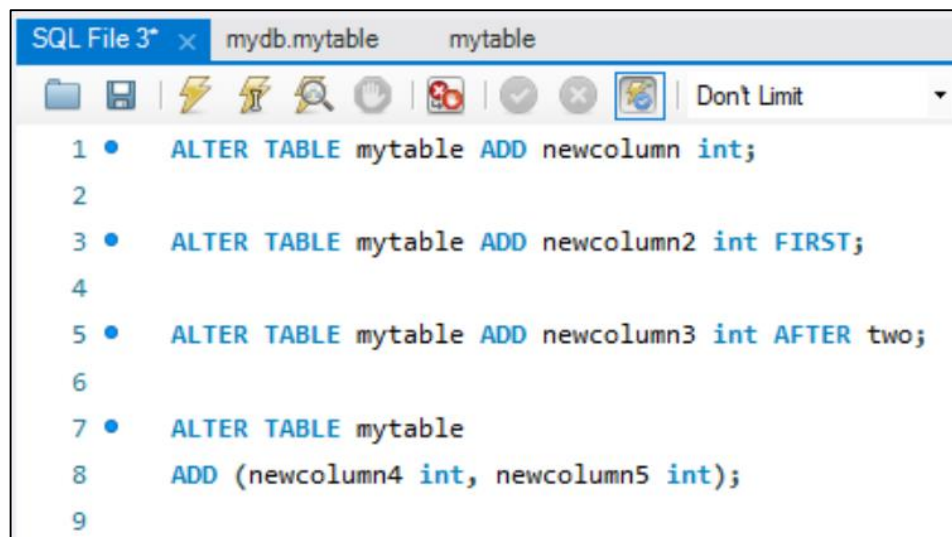
範例

- 下列命令執行結果為何？



```
1 CREATE TABLE mytable (  
2     one int,  
3     two int,  
4     three int  
5 )  
6
```

	one	two	three
--	-----	-----	-------



```
1 ALTER TABLE mytable ADD newcolumn int;  
2  
3 ALTER TABLE mytable ADD newcolumn2 int FIRST;  
4  
5 ALTER TABLE mytable ADD newcolumn3 int AFTER two;  
6  
7 ALTER TABLE mytable  
8 ADD (newcolumn4 int, newcolumn5 int);  
9
```

範例

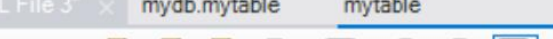
- 下列命令執行結果為何？

```
SQL File 3* x mydb.mytable mytable
[Icons] Don't Limit
1 • ALTER TABLE mytable
2   CHANGE one changecolumn BIGINT AFTER two;
3
```

SQL File 3* mytable mydb.mytable x			
Info Columns Indexes Triggers Foreign keys Partitions Grants			
Column	Type	Default Value	Nullable
newcolumn	int		YES
newcolumn2	int		YES
newcolumn3	int		YES
newcolumn4	int		YES
newcolumn5	int		YES
one	int		YES
three	int		YES
two	int		YES

Result Grid [Icons] Filter Rows: [] Export: [] Wrap Cell Content: []							
newcolumn2	one	two	newcolumn3	three	newcolumn	newcolumn4	newcolumn5

■ 下列命令執行結果為何?



The screenshot shows the SQL File 3* editor with the following content:

```

1 • ALTER TABLE mytable
2   MODIFY changecolumn INT AFTER three;
3

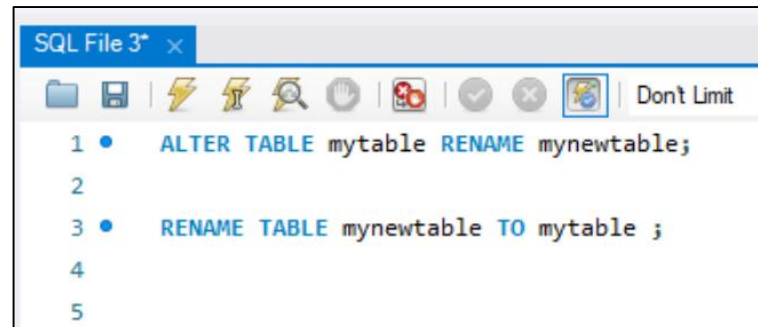
```

Info	Columns	Indexes	Triggers	Foreign keys	Partitions
Column	Type	Default Value	Nullable		
changeColumn	bigint		YES		
newColumn	int		YES		
newColumn2	int		YES		
newColumn3	int		YES		
newColumn4	int		YES		
newColumn5	int		YES		
three	int		YES		
two	int		YES		

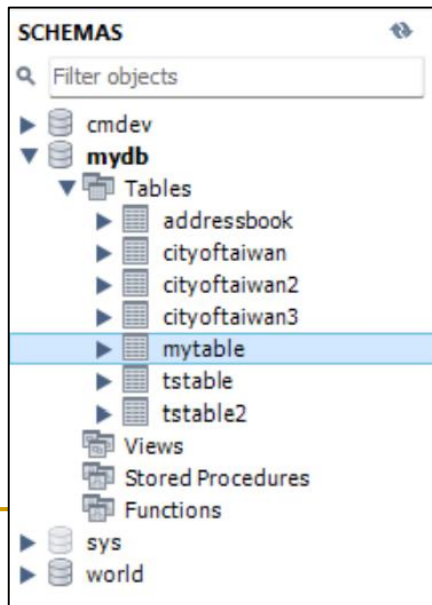
	newcolumn2	two	changecolumn	newcolumn3	three	newcolumn	newcolumn4	newcolumn5
--	------------	-----	--------------	------------	-------	-----------	------------	------------

範例

- 下列命令執行結果為何？



```
SQL File 3* x
1 • ALTER TABLE mytable RENAME mynewtable;
2
3 • RENAME TABLE mynewtable TO mytable ;
4
5
```



查詢表格資訊

- 查詢資料庫包含的表格語法

SHOW TABLES FROM 資料庫名稱 [**LIKE** '字串樣式']

- 查詢系統資訊資料庫 **information_schema**

- **TABLES** 表格：儲存資料庫中表格相關資訊

- **TABLE_SCHEMA** 資料庫名稱
- **TABLE_NAME** 表格名稱
- **ENGINE** 儲存引擎名稱
- **TABLE_ROWS** 紀錄數量
- **TABLE COLLATION** 使用的 collation
- **AUTO_INCREMENT** 自動遞增

查詢表格定義

- 查詢表格定義語法
 - {DESCRIBE|DESC} 表格名稱
 - SHOW {COLUMNS|FIELDS} FROM 表格名稱
 - 取得欄位名稱資料型態及設定屬性
- 查詢建立表格的敘述
 - SHOW CREATE TABLE 表格名稱

範例

- 查詢world資料庫所有表格
- 查詢world資料庫名稱以y結尾表格
- 使用information_schema查詢world資料庫表格資訊

範例

- 查詢world.country表格資訊