

# 資料庫環境簡介

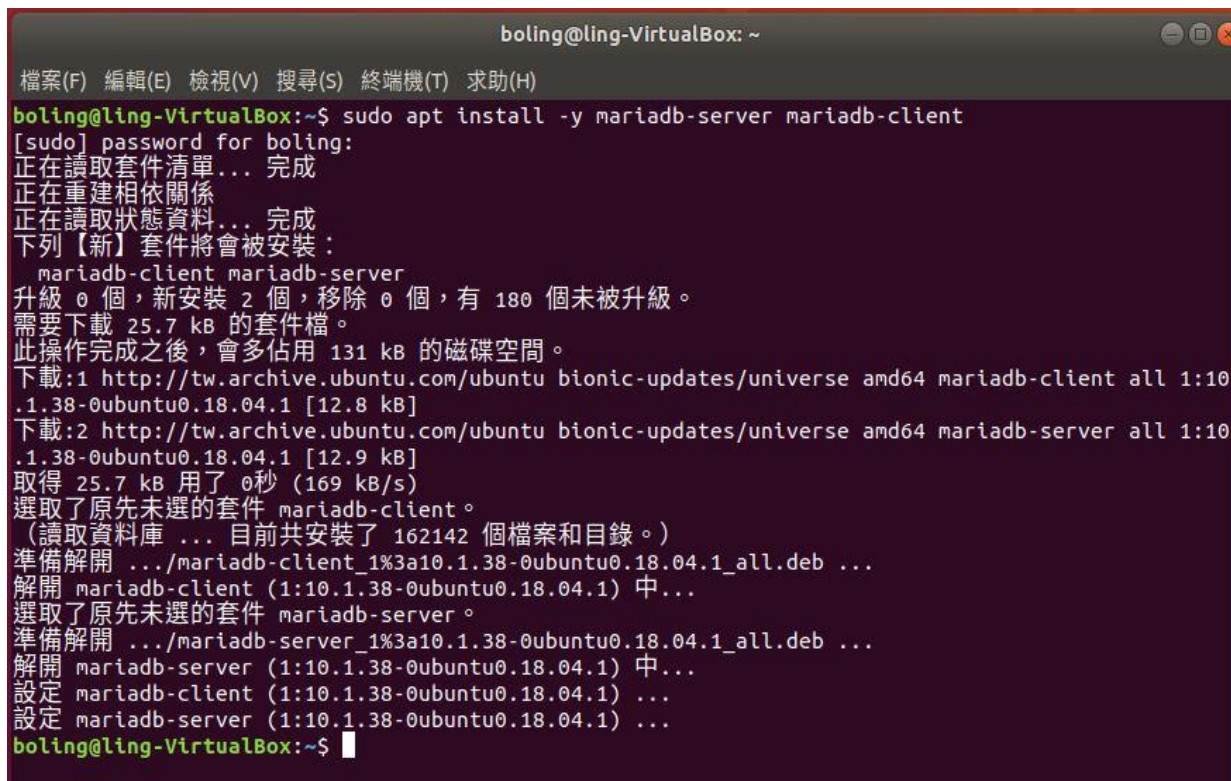


# MySQL 安裝與設定

`sudo apt install -y mariadb-server mariadb-client`

進行mariadb-server的安裝。

指令完成後若出現下圖畫面即為安裝成功。

A terminal window titled 'boling@ling-VirtualBox: ~' showing the execution of the command 'sudo apt install -y mariadb-server mariadb-client'. The output displays the progress of the installation, including package lists, space requirements, and the successful installation of mariadb-client and mariadb-server.

```
boling@ling-VirtualBox: ~  
檔案(F) 編輯(E) 檢視(V) 搜尋(S) 終端機(T) 求助(H)  
boling@ling-VirtualBox:~$ sudo apt install -y mariadb-server mariadb-client  
[sudo] password for boling:  
正在讀取套件清單... 完成  
正在重建相依關係  
正在讀取狀態資料... 完成  
下列【新】套件將會被安裝：  
  mariadb-client mariadb-server  
升級 0 個，新安裝 2 個，移除 0 個，有 180 個未被升級。  
需要下載 25.7 kB 的套件檔。  
此操作完成之後，會多佔用 131 kB 的磁碟空間。  
下載:1 http://tw.archive.ubuntu.com/ubuntu bionic-updates/universe amd64 mariadb-client all 1:10  
.1.38-0ubuntu0.18.04.1 [12.8 kB]  
下載:2 http://tw.archive.ubuntu.com/ubuntu bionic-updates/universe amd64 mariadb-server all 1:10  
.1.38-0ubuntu0.18.04.1 [12.9 kB]  
取得 25.7 kB 用了 0秒 (169 kB/s)  
選取了原先未選的套件 mariadb-client。  
(讀取資料庫 ... 目前共安裝了 162142 個檔案和目錄。)  
準備解開 .../mariadb-client_1%3a10.1.38-0ubuntu0.18.04.1_all.deb ...  
解開 mariadb-client (1:10.1.38-0ubuntu0.18.04.1) 中...  
選取了原先未選的套件 mariadb-server。  
準備解開 .../mariadb-server_1%3a10.1.38-0ubuntu0.18.04.1_all.deb ...  
解開 mariadb-server (1:10.1.38-0ubuntu0.18.04.1) 中...  
設定 mariadb-client (1:10.1.38-0ubuntu0.18.04.1) ...  
設定 mariadb-server (1:10.1.38-0ubuntu0.18.04.1) ...  
boling@ling-VirtualBox:~$
```

# MySQL 安裝與設定

安裝及設定環境完成後，先關閉mariadb。

```
sudo systemctl stop mariadb.service
```

再重新開啟mariadb

```
sudo systemctl start mariadb.service
```

設定mariadb在每次開機後即自動啟動。

```
sudo systemctl enable mariadb.service
```

確認mariadb是否有正常啟動與運作。

```
sudo systemctl status mariadb.service
```

若正常運作即會出現

如圖中Active: active(running)

```
boling@ling-VirtualBox:~$ sudo systemctl stop mariadb.service
boling@ling-VirtualBox:~$ sudo systemctl start mariadb.service
boling@ling-VirtualBox:~$ sudo systemctl enable mariadb.service
boling@ling-VirtualBox:~$ sudo systemctl status mariadb.service
● mariadb.service - MariaDB 10.1.38 database server
   Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2019-05-01 21:56:02 CST; 8s ago
     Docs: man:mysqld(8)
           https://mariadb.com/kb/en/library/systemd/
   Main PID: 10417 (mysqld)
    Status: "Taking your SQL requests now..."
     Tasks: 27 (limit: 4915)
    CGroup: /system.slice/mariadb.service
           └─10417 /usr/sbin/mysqld

5月 01 21:56:02 ling-VirtualBox systemd[1]: Starting MariaDB 10.1.38 database server...
5月 01 21:56:02 ling-VirtualBox mysqld[10417]: 2019-05-01 21:56:02 139720096373888 [Note] /usr/sbin/my
5月 01 21:56:02 ling-VirtualBox /etc/mysql/debian-start[10447]: Upgrading MySQL tables if necessary.
5月 01 21:56:02 ling-VirtualBox systemd[1]: Started MariaDB 10.1.38 database server.
5月 01 21:56:02 ling-VirtualBox /etc/mysql/debian-start[10451]: /usr/bin/mysql_upgrade: the '--basedir
5月 01 21:56:02 ling-VirtualBox /etc/mysql/debian-start[10451]: Looking for 'mysql' as: /usr/bin/mysql
5月 01 21:56:02 ling-VirtualBox /etc/mysql/debian-start[10451]: Looking for 'mysqlcheck' as: /usr/bin/
5月 01 21:56:02 ling-VirtualBox /etc/mysql/debian-start[10451]: This installation of MySQL is already
5月 01 21:56:02 ling-VirtualBox /etc/mysql/debian-start[10459]: Checking for insecure root accounts.
5月 01 21:56:02 ling-VirtualBox /etc/mysql/debian-start[10463]: Triggering myisam-recover for all MyIS
lines 1-21/21 (END)
```

# MySQL 安裝與設定

為使Mysql安全性提高因此進行安全設定。

`sudo mysql_secure_installation`

Enter current password for root (enter for none):

//第一次設定按enter即可。

Set root password?[Y/n]: Y

//是否設定root密碼。

New password:

//輸入欲設定的新密碼。

Re-enter new password:

//再次輸入剛剛輸入的新密碼

```
boling@ling-VirtualBox:~$ sudo mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n] Y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
... Success!
```

# MySQL安裝與設定

Remove anonymous users? [Y/n] Y

//移除匿名用戶。

Disallow root login remotely? [Y/n] Y

//關閉root遠端登入。

Remove test database and access to it? [Y/n] n

//不移除資料表。

Reload privilege tables now? [Y/n] Y

//重新載入資料表的權限

完成以上步驟後重啟MariaDB服務。

```
Remove anonymous users? [Y/n] Y
... Success!

Normally, root should only be allowed to connect from 'localhost'. This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] Y
... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access. This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] n
... skipping.

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] Y
... Success!

Cleaning up...

All done! If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
boling@ling-VirtualBox:~$ sudo systemctl restart mariadb.service
```

# 資料庫系統登入

---

```
sudo mysql -u root -p
```

//-u表示以使用者登入，root代表使用者名稱，-p表示密碼的參數。如果不想密碼被看到，可以直接-p後按enter，將會在下一行要求使用者輸入密碼，並有遮蔽密碼的效果。

```
boling@ling-VirtualBox:~$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.1.38-MariaDB-0ubuntu0.18.04.1 Ubuntu 18.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

# 建立資料庫-1

---

為了在後續介紹資料表的應用，因此我們須先建立一個資料庫。

```
create database securitytest;
```

//新增一個名叫 securitytest 的資料庫。

```
show databases;
```

//顯示所有資料庫資訊，確認資料庫。

```
MariaDB [(none)]> create database securitytest;  
Query OK, 1 row affected (0.00 sec)  
  
MariaDB [(none)]> show databases;  
+-----+  
| Database |  
+-----+  
| information_schema |  
| mysql |  
| performance_schema |  
| securitytest |  
+-----+  
4 rows in set (0.00 sec)
```

# 建立資料庫-2

---

```
use securitytest;
```

//使用 securitytest

意指之後的指令都在這個資料庫執行，其他不會動到，出現Database changed代表已經成功指定在securitytest資料庫下動作。

```
MariaDB [(none)]> use securitytest;  
Database changed  
MariaDB [securitytest]>
```



# 建立資料表(CREATE TABLES)

---

//使用 securitytest 資料庫(資料表才能建立在此當中)。

```
use securitytest;
```

//建立資料表並且設置當中所需要的欄位型態。

```
create table studata(  
stu_Id varchar(5) not null default '00000',  
stu_Name varchar(20) not null default '',  
stu_Sex varchar(2) default 'M',  
stu_Tel varchar(10),  
stu_Mail varchar(50) default 'unknow',  
primary key(stu_Id));
```

```
MariaDB [securitytest]> create table studata(  
-> stu_Id varchar(5) not null default '00000',  
-> stu_Name varchar(20) not null default '',  
-> stu_Sex varchar(2) default 'M',  
-> stu_Tel varchar(10),  
-> stu_Mail varchar(50) default 'unknow',  
-> primary key(stu_Id));  
Query OK, 0 rows affected (0.04 sec)
```

# 檢視資料表 (DESCRIBE TABLES)

---

show tables;

//確認所建立的資料表是否成功。

describe studata;

//顯示剛剛所建立的studata資料表結構。

```
MariaDB [securitytest]> show tables;
+-----+
| Tables_in_securitytest |
+-----+
| studata                  |
+-----+
1 row in set (0.00 sec)

MariaDB [securitytest]> describe studata;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| stu_Id     | varchar(5)    | NO   | PRI | 00000   |       |
| stu_Name   | varchar(20)   | NO   |     |         |       |
| stu_Sex    | varchar(2)    | YES  |     | M       |       |
| stu_Tel    | varchar(10)   | YES  |     | NULL    |       |
| stu_Mail   | varchar(50)   | YES  |     | unknow  |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

# 刪除資料表(DROP TABLES)

為了示範刪除資料表的語法，所以我們須先建立一個名為hellolinux的資料表。  
create table hellolinux(user\_ID int not null default '000');

```
MariaDB [securitytest]> create table hellolinux(user_ID int not null default '000');  
Query OK, 0 rows affected (0.06 sec)
```

接下來我們使用drop tables的語法做刪除資料表的動作。

```
drop tables hellolinux;  
//刪除hellolinux的資料表。  
show tables;  
//確認是否成功。
```

```
MariaDB [securitytest]> drop tables hellolinux;  
Query OK, 0 rows affected (0.05 sec)  
  
MariaDB [securitytest]> show tables;  
+-----+  
| Tables_in_securitytest |  
+-----+  
| studata                 |  
+-----+  
1 row in set (0.00 sec)
```

# 新增資料(INSERT)

---

新增三筆資料至 studata 資料表中。

insert into studata values

(1,'Jack','M','0975730800','nptu123@mail.nptu.edu.tw'),  
(2,'Cindy','F',null,'cindy900@mail.nptu.edu.tw'),  
(3,'David',default,0978654334,'abc123@mail.nptu.edu.tw');

```
MariaDB [securitytest]> insert into studata values  
-> (1,'Jack','M','0975730800','nptu123@mail.nptu.edu.tw'),  
-> (2,'Cindy','F',null,'cindy900@mail.nptu.edu.tw'),  
-> (3,'David',default,0978654334,'abc123@mail.nptu.edu.tw');  
Query OK, 3 rows affected (0.00 sec)  
Records: 3  Duplicates: 0  Warnings: 0
```

# 顯示資料(SELECT)

```
select * from studata;
```

//顯示出studata資料表中所有資料。

```
MariaDB [securitytest]> select * from studata;
```

stu_Id	stu_Name	stu_Sex	stu_Tel	stu_Mail
1	Jack	M	0975730800	nptu123@mail.nptu.edu.tw
2	Cindy	F	NULL	cindy900@mail.nptu.edu.tw
3	David	M	978654334	abc123@mail.nptu.edu.tw

```
3 rows in set (0.00 sec)
```

由上圖可知，因為起初在設定studata資料表結構時將stu\_Sex欄位設有預設值M，所以想使用預設值的話可以直接於insert資料中輸入default，如第三筆。再者，觀察第一筆與第三筆資料的stu\_Tel之差異，可發現於insert時沒有將資料使用單引號(')，會使得電腦將資料誤判為數字而不是一段文字，所以將首字0去除，以下將會介紹使用修改的語法將此資料做修正。

# 修改資料(UPDATE)

修改studata資料表中的stu\_Tel 欄位資料，當stu\_Id欄位值為3時stu\_Tel=0978654334。

```
update studata
```

```
set stu_Tel ='0978654334' where stu_Id=3;
```

```
select * from studata;
```

//確認是否有成功修改的資料。

```
MariaDB [securitytest]> update studata
-> set stu_Tel ='0978654334' where stu_Id=3;
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [securitytest]> select * from studata;
+-----+-----+-----+-----+-----+
| stu_Id | stu_Name | stu_Sex | stu_Tel   | stu_Mail                |
+-----+-----+-----+-----+-----+
| 1      | Jack    | M      | 0975730800 | nptu123@mail.nptu.edu.tw |
| 2      | Cindy   | F      | NULL       | cindy900@mail.nptu.edu.tw |
| 3      | David   | M      | 0978654334 | abc123@mail.nptu.edu.tw  |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

# 清空資料(TRUNCATE)-1

首先建立stu\_score資料表並新增五筆資料作為後續示範使用。

Step1. create table stu\_score(

stu\_Id varchar(5) not null default '00000',

english int,

math int,

chinese int,

primary key(stu\_Id));

```
MariaDB [securitytest]> describe stu_score;
+-----+-----+-----+-----+-----+
| Field | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| stu_Id | varchar(5) | NO   | PRI | 00000   |       |
| english | int(11)    | YES  |     | NULL    |       |
| math   | int(11)    | YES  |     | NULL    |       |
| chinese | int(11)    | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

Step2. insert into stu\_score values

(1,65,70,55),

(2,94,49,74),

(3,83,58,76),

(4,65,92,46);

```
MariaDB [securitytest]> select * from stu_score;
+-----+-----+-----+-----+
| stu_Id | english | math | chinese |
+-----+-----+-----+-----+
| 1      | 65      | 70   | 55      |
| 2      | 94      | 49   | 74      |
| 3      | 83      | 58   | 76      |
| 4      | 65      | 92   | 46      |
+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

# 清空資料(TRUNCATE)-2

```
select * from stu_score;
```

//顯示資料表的資料。

```
truncate table stu_score;
```

//清空資料表中的所有資料。

```
select * from stu_score;
```

//確認資料是否已被清除。

```
MariaDB [securitytest]> select * from stu_score;
+-----+-----+-----+-----+
| stu_Id | english | math | chinese |
+-----+-----+-----+-----+
| 1      | 65      | 70   | 55      |
| 2      | 94      | 49   | 74      |
| 3      | 83      | 58   | 76      |
| 4      | 65      | 92   | 46      |
+-----+-----+-----+-----+
4 rows in set (0.00 sec)

MariaDB [securitytest]> truncate table stu_score;
Query OK, 0 rows affected (0.11 sec)

MariaDB [securitytest]> select * from stu_score;
Empty set (0.01 sec)
```



# 資料排序 (ORDER BY)

`select * from studata order by stu_Name desc;`

//stu\_Name 由ASCII碼做大到小排序。

`select * from studata order by stu_Name asc;`

//stu\_Name 由ASCII碼做小到大排序。

```
MariaDB [securitytest]> select * from studata order by stu_Name desc;
+-----+-----+-----+-----+-----+
| stu_Id | stu_Name | stu_Sex | stu_Tel   | stu_Mail                               |
+-----+-----+-----+-----+-----+
| 1      | Jack    | M      | 0975730800 | nptu123@mail.nptu.edu.tw             |
| 3      | David   | M      | 0978654334 | abc123@mail.nptu.edu.tw               |
| 2      | Cindy   | F      | NULL      | cindy900@mail.nptu.edu.tw             |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

MariaDB [securitytest]> select * from studata order by stu_Name asc;
+-----+-----+-----+-----+-----+
| stu_Id | stu_Name | stu_Sex | stu_Tel   | stu_Mail                               |
+-----+-----+-----+-----+-----+
| 2      | Cindy   | F      | NULL      | cindy900@mail.nptu.edu.tw             |
| 3      | David   | M      | 0978654334 | abc123@mail.nptu.edu.tw               |
| 1      | Jack    | M      | 0975730800 | nptu123@mail.nptu.edu.tw             |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

# 資料表新增欄位

```
alter table studata add column updatetime timestamp default  
current_timestamp on update current_timestamp;  
//新增updatetime 欄位作為紀錄資料被更新時間。
```

```
MariaDB [securitytest]> alter table studata add column updatetime timestamp default current_timestamp  
on update current_timestamp;  
Query OK, 0 rows affected (0.11 sec)  
Records: 0 Duplicates: 0 Warnings: 0
```

```
select * from studata;  
//確認是否有成功新增updatetime 欄位，並且查看資料。
```

```
MariaDB [securitytest]> select * from studata;  
+-----+-----+-----+-----+-----+-----+  
| stu_Id | stu_Name | stu_Sex | stu_Tel   | stu_Mail                               | updatetime           |  
+-----+-----+-----+-----+-----+-----+  
| 1      | Jack     | M       | 0975730800 | nptu123@mail.nptu.edu.tw             | 2019-05-01 23:06:47 |  
| 2      | Cindy    | F       | NULL       | cindy900@mail.nptu.edu.tw            | 2019-05-01 23:06:47 |  
| 3      | David    | M       | 0978654334 | abc123@mail.nptu.edu.tw               | 2019-05-01 23:06:47 |  
+-----+-----+-----+-----+-----+-----+  
3 rows in set (0.01 sec)
```

# 聚合函數-1

為了以下示範先新增score欄位做使用。

```
alter table studata add column score int;
```

//新增欄位score於studata資料表中。

```
update studata set score=46 where stu_Id=1;
```

//新增資料於學生1當中。

```
update studata set score=94 where stu_Id=2;
```

//新增資料於學生2當中。

```
update studata set score=81 where stu_Id=3;
```

//新增資料於學生3當中。

```
MariaDB [securitytest]> alter table studata add column score int;
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0

MariaDB [securitytest]> update studata set score=46 where stu_Id=1;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [securitytest]> update studata set score=94 where stu_Id=2;
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [securitytest]> update studata set score=81 where stu_Id=3;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [securitytest]> select * from studata;
```

stu_Id	stu_Name	stu_Sex	stu_Tel	stu_Mail	updatetime	score
1	Jack	M	0975730800	nptu123@mail.nptu.edu.tw	2019-05-01 23:09:02	46
2	Cindy	F	NULL	cindy900@mail.nptu.edu.tw	2019-05-01 23:09:15	94
3	David	M	0978654334	abc123@mail.nptu.edu.tw	2019-05-01 23:09:24	81

```
3 rows in set (0.00 sec)
```

# 聚合函數-2

select [欲使用的函數(參考欄位)] as ['顯示的欄位名稱'] from [資料表名稱];

select count(\*) as '筆數',

max(score) as '最高分數',

min(score) as '最低分數',

avg(score) as '平均分數',

sum(score) as '總分' from studata;

\*count-->計算資料數量，max-->該欄位資料最大值，min-->該欄位資料最小值，avg-->欄位資料平均值，sum-->欄位資料值的總和。

```
MariaDB [securitytest]> select count(*) as '筆數',  
-> max(score) as '最高分數',  
-> min(score) as '最低分數',  
-> avg(score) as '平均分數',  
-> sum(score) as '總分' from studata;
```

筆數	最高分數	最低分數	平均分數	總分
3	94	46	73.6667	221

```
1 row in set (0.00 sec)
```

# 合併顯示資料表

Step1. 建立stu\_score資料表作為後續示範使用。

```
create table stu_score(
```

```
    stu_Id varchar(5) not null default '00000',
```

```
    english int,
```

```
    math int,
```

```
    chinese int,
```

```
    primary key(stu_Id));
```

```
MariaDB [securitytest]> describe stu_score;
```

Field	Type	Null	Key	Default	Extra
stu_Id	varchar(5)	NO	PRI	00000	
english	int(11)	YES		NULL	
math	int(11)	YES		NULL	
chinese	int(11)	YES		NULL	

```
4 rows in set (0.00 sec)
```

Step2. 新增五筆資料至stu\_score。

```
insert into stu_score values
```

```
    (1,65,70,55),
```

```
    (2,94,49,74),
```

```
    (3,83,58,76),
```

```
    (4,65,92,46);
```

```
MariaDB [securitytest]> select * from stu_score;
```

stu_Id	english	math	chinese
1	65	70	55
2	94	49	74
3	83	58	76
4	65	92	46

```
4 rows in set (0.00 sec)
```

# 卡氏積(又稱交叉乘積、交叉合併)

select \* from 資料表1,資料表2;

select \* from studata,stu\_score;

//同時顯示出兩個資料表(studata、stu\_score)的資料，顯示資料時的資料數量為資料表1的a筆資料\*資料表2的b筆資料。

```
MariaDB [securitytest]> select * from studata,stu_score;
```

stu_Id	stu_Name	stu_Sex	stu_Tel	stu_Mail	updatetime	score	stu_Id	english	math	chinese
1	Jack	M	0975730800	nptu123@mail.nptu.edu.tw	2019-05-01 23:09:02	46	1	65	70	55
2	Cindy	F	NULL	cindy900@mail.nptu.edu.tw	2019-05-01 23:09:15	94	1	65	70	55
3	David	M	0978654334	abc123@mail.nptu.edu.tw	2019-05-01 23:09:24	81	1	65	70	55
1	Jack	M	0975730800	nptu123@mail.nptu.edu.tw	2019-05-01 23:09:02	46	2	94	49	74
2	Cindy	F	NULL	cindy900@mail.nptu.edu.tw	2019-05-01 23:09:15	94	2	94	49	74
3	David	M	0978654334	abc123@mail.nptu.edu.tw	2019-05-01 23:09:24	81	2	94	49	74
1	Jack	M	0975730800	nptu123@mail.nptu.edu.tw	2019-05-01 23:09:02	46	3	83	58	76
2	Cindy	F	NULL	cindy900@mail.nptu.edu.tw	2019-05-01 23:09:15	94	3	83	58	76
3	David	M	0978654334	abc123@mail.nptu.edu.tw	2019-05-01 23:09:24	81	3	83	58	76
1	Jack	M	0975730800	nptu123@mail.nptu.edu.tw	2019-05-01 23:09:02	46	4	65	92	46
2	Cindy	F	NULL	cindy900@mail.nptu.edu.tw	2019-05-01 23:09:15	94	4	65	92	46
3	David	M	0978654334	abc123@mail.nptu.edu.tw	2019-05-01 23:09:24	81	4	65	92	46

12 rows in set (0.00 sec)

# 對等合併(Equi-Join)

select \* from 資料表1,資料表2 where 條件;

select \* from studata,stu\_score where studata.stu\_Id=stu\_score.stu\_Id;

```
MariaDB [securitytest]> select * from studata,stu_score where studata.stu_Id=stu_score.stu_Id;
```

stu_Id	stu_Name	stu_Sex	stu_Tel	stu_Mail	updatetime	score	stu_Id	english	math	chinese
1	Jack	M	0975730800	nptu123@mail.nptu.edu.tw	2019-05-01 23:09:02	46	1	65	70	55
2	Cindy	F	NULL	cindy900@mail.nptu.edu.tw	2019-05-01 23:09:15	94	2	94	49	74
3	David	M	0978654334	abc123@mail.nptu.edu.tw	2019-05-01 23:09:24	81	3	83	58	76

```
3 rows in set (0.00 sec)
```

# 左、右外部合併

- 左外部合併

select \* from 資料表1 left outer join 資料表2 on 資料表1.[欄位]=資料表2.[欄位];

select \* from studata left outer join stu\_score on studata.stu\_Id=stu\_score.stu\_Id;

```
MariaDB [securitytest]> select * from studata left outer join stu_score on studata.stu_Id=stu_score.stu_Id;
```

stu_Id	stu_Name	stu_Sex	stu_Tel	stu_Mail	updatetime	score	stu_Id	english	math	chinese
1	Jack	M	0975730800	nptu123@mail.nptu.edu.tw	2019-05-01 23:09:02	46	1	65	70	55
2	Cindy	F	NULL	cindy900@mail.nptu.edu.tw	2019-05-01 23:09:15	94	2	94	49	74
3	David	M	0978654334	abc123@mail.nptu.edu.tw	2019-05-01 23:09:24	81	3	83	58	76

3 rows in set (0.00 sec)

- 右外部合併

select \* from 資料表1 right join 資料表2 on 資料表1.[欄位]=資料表2.[欄位];

select \* from studata right join stu\_score on studata.stu\_Id=stu\_score.stu\_Id;

```
MariaDB [securitytest]> select * from studata right join stu_score on studata.stu_Id=stu_score.stu_Id;
```

stu_Id	stu_Name	stu_Sex	stu_Tel	stu_Mail	updatetime	score	stu_Id	english	math	chinese
1	Jack	M	0975730800	nptu123@mail.nptu.edu.tw	2019-05-01 23:09:02	46	1	65	70	55
2	Cindy	F	NULL	cindy900@mail.nptu.edu.tw	2019-05-01 23:09:15	94	2	94	49	74
3	David	M	0978654334	abc123@mail.nptu.edu.tw	2019-05-01 23:09:24	81	3	83	58	76
NULL	NULL	NULL	NULL	NULL	NULL	NULL	4	65	92	46

4 rows in set (0.00 sec)



# MySQL 加密

新增一個名為Userinfo的資料表作以下MySQL加密示範使用。

```
create table Userinfo (  
    user_SN int(10) not null auto_increment,  
    user_ID varchar(20) not null,  
    user_PW varbinary(255),  
    user_Name varchar(20) not null,  
    time timestamp not null default current_timestamp,  
    primary key(user_SN),  
    unique key (user_ID));
```

```
MariaDB [securitytest]> describe Userinfo;
```

Field	Type	Null	Key	Default	Extra
user_SN	int(10)	NO	PRI	NULL	auto_increment
user_ID	varchar(20)	NO	UNI	NULL	
user_PW	varbinary(255)	YES		NULL	
user_Name	varchar(20)	NO		NULL	
time	timestamp	NO		CURRENT_TIMESTAMP	

5 rows in set (0.00 sec)

# 雙向加密-Encode()-1

encode('密碼','金鑰')

先新增兩筆資料進入資料表中，如下圖。

1. insert into Userinfo(user\_ID,user\_PW,user\_Name)  
values('abc123','qwe456','周杰倫');
2. insert into Userinfo(user\_ID,user\_PW,user\_Name)  
values('def654','qcv54w','楊陳林');

```
MariaDB [securitytest]> select * from Userinfo;
```

user_SN	user_ID	user_PW	user_Name	time
1	abc123	qwe456	周杰倫	2019-05-01 23:30:17
2	def654	qcv54w	楊陳林	2019-05-01 23:30:41

```
2 rows in set (0.00 sec)
```

# 雙向加密-Encode()-2

- 利用update方法將user\_PW使用encode()加密。

update Userinfo set user\_PW=encode('qwe456','test1') where user\_SN=1;

update Userinfo set user\_PW=encode('red4v63','test1') where user\_SN=2;

```
MariaDB [securitytest]> select * from Userinfo;
```

user_SN	user_ID	user_PW	user_Name	time
1	abc123	◆◆^.Gb	周杰倫	2019-05-01 23:30:17
2	def654	tqq>41◆	楊陳林	2019-05-01 23:30:41

```
2 rows in set (0.00 sec)
```

- 利用insertz方法新增user\_PW時使用encode()加密。

insert into Userinfo(user\_ID,user\_PW,user\_Name) values

('tgh963da',encode('rtg964da','test1'),'蕭勁藤');

```
MariaDB [securitytest]> select * from Userinfo;
```

user_SN	user_ID	user_PW	user_Name	time
1	abc123	◆◆^.Gb	周杰倫	2019-05-01 23:30:17
2	def654	tqq>41◆	楊陳林	2019-05-01 23:30:41
3	tgh963da	t◆◆>~◆~◆	蕭勁藤	2019-05-01 23:34:37

```
3 rows in set (0.00 sec)
```

# 雙向加密-Decode()

使用decode方法將加密過後的密碼解密。

decode(欄位名稱,'金鑰')

select \*, decode(user\_PW,'test1') as decode from Userinfo;

```
MariaDB [securitytest]> select * , decode(user_PW,'test1') as decode from Userinfo;
```

user_SN	user_ID	user_PW	user_Name	time	decode
1	abc123	^^^.Gb	周杰倫	2019-05-01 23:30:17	qwe456
2	def654	tqq>41^	楊陳林	2019-05-01 23:30:41	red4v63
3	tgh963da	t^^>~^^	蕭勁滕	2019-05-01 23:34:37	rtg964da

3 rows in set (0.01 sec)

# 雙向加密-AES\_ENCRYPT()

`aes_encrypt('密碼', '金鑰')`

利用insert 資料於資料表時將密碼使用[aes\\_encrypt\(\)](#)加密。

1. `insert into Userinfo(user_ID,user_PW,user_Name)`  
`values('wec5413',aes_encrypt('okg9621','test2'),'林又家');`
2. `insert into Userinfo(user_ID,user_PW,user_Name)`  
`values('ked963',aes_encrypt('lkm85daw61','test2'),'莫雯尉');`

```
MariaDB [securitytest]> select * from Userinfo;
```

user_SN	user_ID	user_PW	user_Name	time
1	abc123	♦♦^.Gb	周杰倫	2019-05-01 23:30:17
2	def654	tqq>41♦	楊陳林	2019-05-01 23:30:41
3	tgh963da	t♦♦>~♦~♦	蕭勁藤	2019-05-01 23:34:37
4	wec5413	♦♦Y♦}♦~♦[09]♦♦♦Opc	林又家	2019-05-01 23:41:29
5	ked963	♦lw!Z[09]_}♦[09]♦7♦	莫雯尉	2019-05-01 23:41:44

```
5 rows in set (0.00 sec)
```

# 雙向加密-AES\_DECRYPT()

`aes_decrypt(欄位,'金鑰')`

將user\_PW使用aes\_decrypt()且金鑰是test2的密碼解密。

`select * , aes_decrypt(user_PW,'test2') as aes_decrypt from Userinfo where user_SN=4 OR user_SN=5;`

```
MariaDB [securitytest]> select * , aes_decrypt(user_PW,'test2') as aes_decrypt from Userinfo where user_SN=4 OR user_SN=5;
```

user_SN	user_ID	user_PW	user_Name	time	aes_decrypt
4	wec5413	Y~0pc	林又家	2019-05-01 23:41:29	okg9621
5	ked963	lw!Z_}7	莫雯尉	2019-05-01 23:41:44	lkm85daw61

2 rows in set (0.00 sec)

# 雙向加密-DES\_ENCRYPT()

`des_encrypt('密碼', '金鑰')`

利用insert 資料於資料表時將密碼使用`des_encrypt()`加密。

1. `insert into Userinfo(user_ID,user_PW,user_Name)`  
`values('96weq85',des_encrypt('63gwes84','test3'),'王大福');`
2. `insert into Userinfo(user_ID,user_PW,user_Name)`  
`values('85fsw',des_encrypt('vbn3669','test3'),'張卿坊');`

```
MariaDB [securitytest]> select * from Userinfo;
```

user_SN	user_ID	user_PW	user_Name	time
1	abc123	^Gb	周杰倫	2019-05-01 23:30:17
2	def654	tqq>41	楊陳林	2019-05-01 23:30:41
3	tgh963da	t>~>	蕭勁藤	2019-05-01 23:34:37
4	wec5413	Y}~>0pc	林又家	2019-05-01 23:41:29
5	ked963	lw!Z _}7	莫雯尉	2019-05-01 23:41:44
6	96weq85			
7	85fsw	z>t	張卿坊	2019-05-01 23:45:37

7 rows in set (0.00 sec)

# 雙向加密-DES\_DECRYPT()

des\_decrypt(欄位,'金鑰')

將user\_PW使用des\_decrypt()且是金鑰test2的密碼解密。

```
select * , des_decrypt(user_PW,'test3') as des_decrypt from Userinfo  
where user_SN=6 OR user_SN=7;
```

```
MariaDB [securitytest]> select * , des_decrypt(user_PW,'test3') as des_decrypt from Userinfo where user_SN=6 OR user_SN=7;  
+-----+-----+-----+-----+-----+-----+  
| user_SN | user_ID | user_PW          | user_Name | time           | des_decrypt |  
+-----+-----+-----+-----+-----+-----+  
| 6       | 96weq85 | 王 大 福        | 王大福   | 2019-05-01 23:45:28 | 63gwes84    |  
yl 1;/R   | 王大福   | 2019-05-01 23:45:28 | 63gwes84 |  
| 7       | 85fsw   | 張 卿 坊        | 張卿坊   | 2019-05-01 23:45:37 | vbn3669     |  
+-----+-----+-----+-----+-----+-----+  
2 rows in set (0.00 sec)
```



# 單向加密-MD5()

md5('密碼')

利用insert 資料於資料表時將密碼使用MD5()加密。

```
insert into Userinfo(user_ID,user_PW,user_Name)
values('cef86314',md5('ty9621w'),'周興哲');
```

```
MariaDB [securitytest]> select * from Userinfo where user_SN=8;
```

user_SN	user_ID	user_PW	user_Name	time
8	cef86314	47d923619176583b13ae7c92d37cf553	周興哲	2019-05-01 23:49:48

```
1 row in set (0.00 sec)
```

由於此方法為單向加密，因此無法將加密後的密碼還原，但仍可以用count去比對加密前後的密碼是否匹配。

```
select count(*) from Userinfo where user_SN=8 and
user_PW=md5('ty9621w');
```

```
MariaDB [securitytest]> select count(*) from Userinfo where user_SN=8 and user_PW=md5('ty9621w');
```

count(*)
1

```
1 row in set (0.00 sec)
```

# 單向加密-PASSWORD()

password('密碼')

利用insert 資料於資料表時將密碼使用PASSWORD()加密。

insert into Userinfo(user\_ID,user\_PW,user\_Name)

```
MariaDB [securitytest]> select * from Userinfo where user_SN=9;
```

user_SN	user_ID	user_PW	user_Name	time
9	asd567	*5D2E4A16670726F1EE88E8345189BD9C712DA478	茄子蛋	2019-05-01 23:54:52

```
1 row in set (0.00 sec)
```

一樣使用count比對加密後的密碼，顯示結果如下圖。

select count(\*) from Userinfo where user\_SN=9 and  
user\_PW=password('qwdf234');

```
MariaDB [securitytest]> select count(*) from Userinfo where user_SN=9 and user_PW=password('qwdf234');
```

count(*)
1

```
1 row in set (0.00 sec)
```

# 單向加密-ENCRYPT()

---

encrypt('密碼', '金鑰')

利用insert 資料於資料表時將密碼使用ENCRYPT()加密。

```
insert into Userinfo(user_ID,user_PW,user_Name)
values('olm84k',encrypt('yui8413','testencrypt'),'蔡一霖');
```

```
MariaDB [securitytest]> select * from Userinfo where user_SN=10;
+-----+-----+-----+-----+-----+
| user_SN | user_ID | user_PW      | user_Name | time                |
+-----+-----+-----+-----+-----+
|      10 | olm84k  | tenWCq/ayHJZc | 蔡一霖    | 2019-05-01 23:58:16 |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

# 單向加密-SHA()、SHA1()

sha('密碼') or sha1('密碼')

SHA()&SHA1()方法是一樣的所以本範例採用SHA()加密。

利用insert 資料於資料表時將密碼使用SHA()加密。

insert into Userinfo(user\_ID,user\_PW,user\_Name)

```
MariaDB [securitytest]> select * from Userinfo where user_SN=11;
```

user_SN	user_ID	user_PW	user_Name	time
11	85osk9	fe344d892d812fcdcb5a8488087a7a79aca9a280	張匯妹	2019-05-01 23:59:53

1 row in set (0.00 sec)

使用count比對加密後的密碼，顯示結果如下圖。

select count(\*) from Userinfo where user\_SN=11 and  
user\_PW=sha('854ewwe');

```
MariaDB [securitytest]> select count(*) from Userinfo where user_SN=11 and user_PW=sha('854ewwe');  
+-----+  
| count(*) |  
+-----+  
| 1 |  
+-----+  
1 row in set (0.00 sec)
```