資料庫環境簡介

屏東大學資管系教授

陳俊麟

MySQL安裝與設定

在進行系統安裝更新前須先登入superuser。

1. su //以 root 權限做一些更改檔案等小動作
2. su -

//進行比較複雜的系統管理，牽涉到許多 root 帳號的環境變數（例如 PATH 或 MAIL 等）

```
[boling@localhost ~]$ su -
密碼：
```

3. sudo su

//用來取得 root 或是其他帳號的權限，不過它在取得 root 或其他帳號權限的時候，是輸入自己的密碼，而不是 root 或其他帳號的密碼

密碼: Aa123456

- dnf -y install mariadb-server
- 以上指令完成後若出現下圖畫面即為安裝成功。

```
已安裝：
  mariadb-server.x86_64 3:10.1.33-1.fc26
  libsphinxclient.x86_64 2.2.11-3.fc26
  mariadb-server-utils.x86_64 3:10.1.33-1.fc26
  sphinx.x86_64 2.2.11-3.fc26
  bison.x86_64 3.0.4-6.fc26
  jemalloc.x86_64 4.5.0-5.fc26
  mariadb.x86_64 3:10.1.33-1.fc26
  mariadb-errmsg.x86_64 3:10.1.33-1.fc26
  perl-DBD-MySQL.x86_64 4.043-1.fc26
  perl-DBI.x86_64 1.636-4.fc26
  perl-Math-BigInt.noarch 1.9998.11-1.fc26
  perl-Math-Complex.noarch 1.59-397.fc26
  perl-Storable.x86_64 1:2.56-368.fc26

完成！
```

MySQL安裝與設定

為了預防之後使用中文時會出現編譯上的亂碼，因此進入

etc/my.cnf.d/mariadb-server.cnf

第21行的修改，如下圖所示。

vi /etc/my.cnf.d/mariadb-server.cnf

=>進入指令模式後，按Shift+， 出現:時，輸入21(表示搜尋21行)

=>按i進入編輯模式，於21行輸入character-set-server=utf8

=>輸入完成後按Esc結束編輯模式

=>進入指令模式後，按Shift+ ，出現:時，輸入wq儲存並離開

```
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
log-error=/var/log/mariadb/mariadb.log
pid-file=/run/mariadb/mariadb.pid
character-set-server=utf8
```

MySQL安裝與設定

安裝及設定環境完成後，開啟mariadb。

systemctl start mariadb

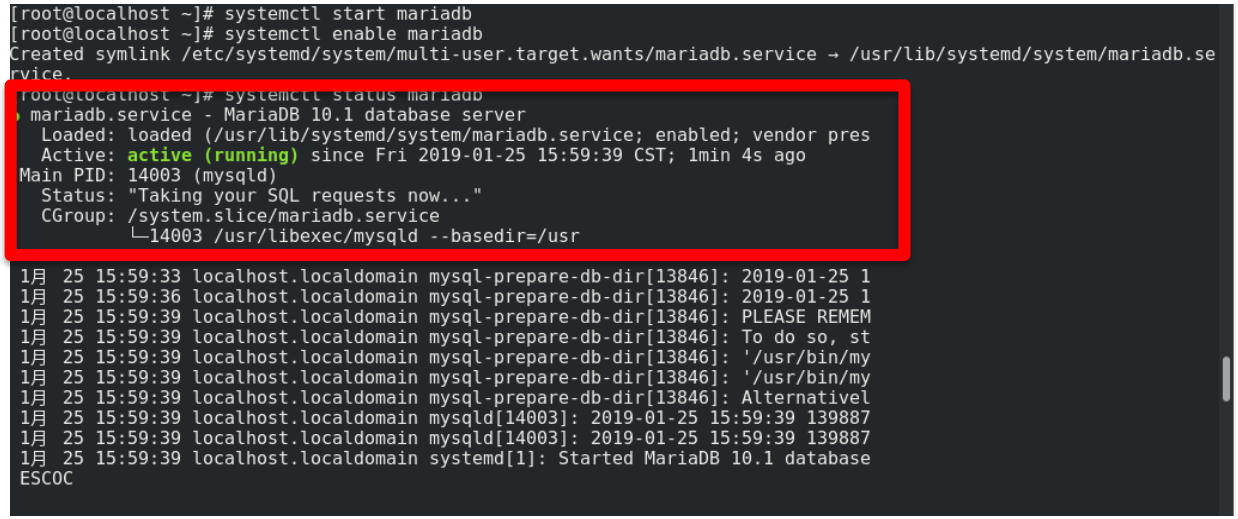
設定mariadb在每次開機後即自動啟動。

systemctl enable mariadb

確認mariadb是否有正常啟動與運作。

systemctl status mariadb

若正常運作即會出現如圖中Active: active(running)。



MySQL安裝與設定

為使Mysql安全性提高因此進行安全設定。

mysql_secure_installation

Enter current password for root (enter for none):

//第一次設定按enter即可。

Switch to unix_socket authentication [Y/n] n

…

Change the root password? [Y/n] n

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

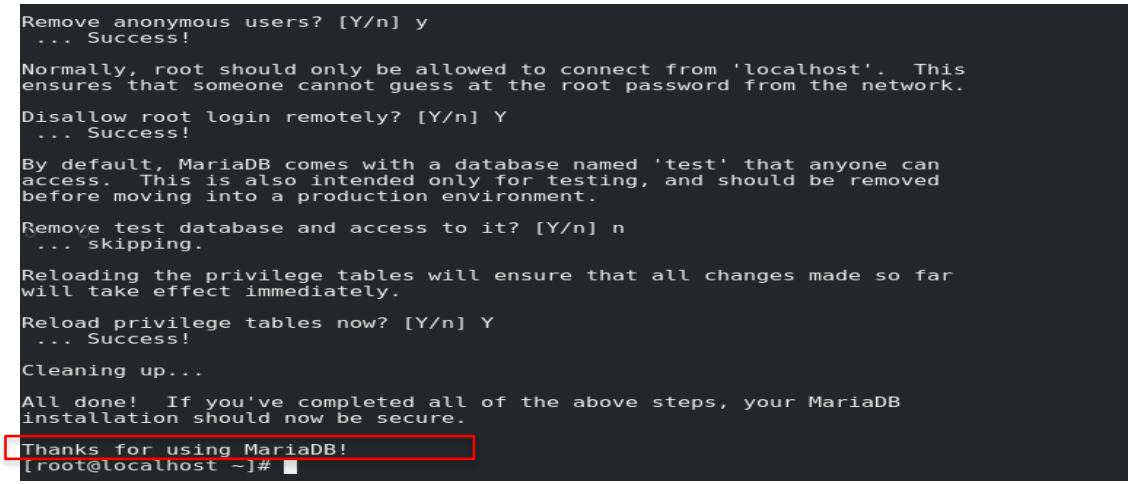
//不移除資料表。

Reload privilege tables now? [Y/n] Y

//重新載入資料表的權限

完成以上步驟出現如右圖

紅框中的文字表示設定成功



資料庫系統登入

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@localhost ~]$ mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 3
Server version: 10.1.33-MariaDB MariaDB Server

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

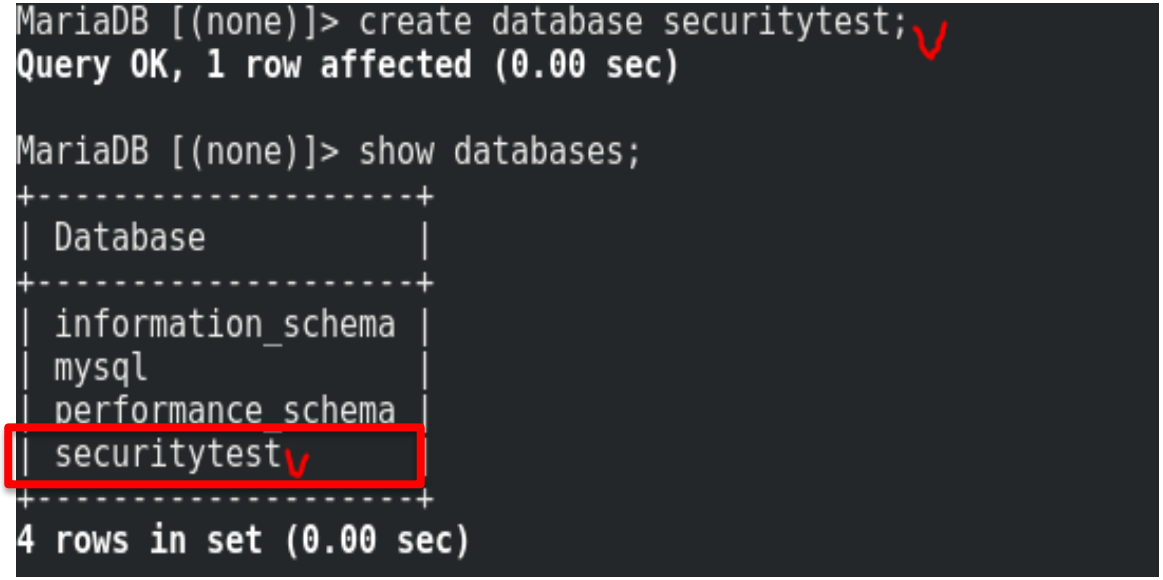
為了在後續介紹資料表的應用，因此我們須先建立一個資料庫。

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。



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 [(none)]> use securitytest
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
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 |
+------------------------+
| hellolinux             |
| studata                |
+------------------------+
2 rows 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 (0.04 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 |
+-----------------------+
| Userinfo              |
| stu_score             |
| studata               |
+-----------------------+
3 rows 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 (0.00 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爲後續示範使用。

Step1. create table stu_score(

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

english int,

math int,

chinese int,

primary key(stu_Id));

```
+---------+------------+------+-----+---------+-------+
| 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);

```
+--------+---------+------+---------+
| 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 (0.01 sec)

MariaDB [securitytest]> truncate table stu_score;
Query OK, 0 rows affected (0.04 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_timest
amp on update current_timestamp;
Query OK, 0 rows affected (0.05 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 | 2019-01-30 16:10:39 |
| 2      | Cindy    | F       | NULL       | cindy900@mail.nptu.edu.tw | 2019-01-30 16:10:39 |
| 3      | David    | M       | 0978654334 | abc123@mail.nptu.edu.tw  | 2019-01-30 16:10:39 |
+--------+----------+---------+------------+--------------------------+---------------------+
3 rows in set (0.00 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]> update studata set score=46 where stu_Id=1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [securitytest]> update studata set score=94 where stu_Id=2;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [securitytest]> update studata set score=81 where stu_Id=3;
Query OK, 1 row affected (0.01 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 | 2019-02-18 14:53:21 |    46 |
| 2      | Cindy    | F       | NULL       | cindy900@mail.nptu.edu.tw | 2019-02-18 14:53:45 |    94 |
| 3      | David    | M       | 0978654334 | abc123@mail.nptu.edu.tw | 2019-02-18 14:54:00 |    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 (0.01 sec)
```

合併顯示資料表

Step1. 建立stu_score資料表作爲後續示範使用。

```
create table stu_score(
        stu_Id varchar(5) not null default '00000',
        english int,
        math int,
        chinese int,
        primary key(stu_Id));
```

```
+---------+------------+------+-----+---------+-------+
| 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);
```

```
+--------+---------+------+---------+
| 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 | 2019-01-30 16:24:53 |    46 | 1      |      65 |   70 |      55 |
| 2      | Cindy    | F       | NULL       | cindy900@mail.nptu.edu.tw | 2019-01-30 16:25:17 |    94 | 1      |      65 |   70 |      55 |
| 3      | David    | M       | 0978654334 | abc123@mail.nptu.edu.tw  | 2019-01-30 16:25:31 |    81 | 1      |      65 |   70 |      55 |
| 1      | Jack     | M       | 0975730800 | nptu123@mail.nptu.edu.tw | 2019-01-30 16:24:53 |    46 | 2      |      94 |   49 |      74 |
| 2      | Cindy    | F       | NULL       | cindy900@mail.nptu.edu.tw | 2019-01-30 16:25:17 |    94 | 2      |      94 |   49 |      74 |
| 3      | David    | M       | 0978654334 | abc123@mail.nptu.edu.tw  | 2019-01-30 16:25:31 |    81 | 2      |      94 |   49 |      74 |
| 1      | Jack     | M       | 0975730800 | nptu123@mail.nptu.edu.tw | 2019-01-30 16:24:53 |    46 | 3      |      83 |   58 |      76 |
| 2      | Cindy    | F       | NULL       | cindy900@mail.nptu.edu.tw | 2019-01-30 16:25:17 |    94 | 3      |      83 |   58 |      76 |
| 3      | David    | M       | 0978654334 | abc123@mail.nptu.edu.tw  | 2019-01-30 16:25:31 |    81 | 3      |      83 |   58 |      76 |
| 1      | Jack     | M       | 0975730800 | nptu123@mail.nptu.edu.tw | 2019-01-30 16:24:53 |    46 | 4      |      65 |   92 |      46 |
| 2      | Cindy    | F       | NULL       | cindy900@mail.nptu.edu.tw | 2019-01-30 16:25:17 |    94 | 4      |      65 |   92 |      46 |
| 3      | David    | M       | 0978654334 | abc123@mail.nptu.edu.tw  | 2019-01-30 16:25:31 |    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 | 2019-01-30 16:24:53 |    46 | 1      |      65 |   70 |      55 |
| 2      | Cindy    | F      | NULL       | cindy900@mail.nptu.edu.tw | 2019-01-30 16:25:17 |    94 | 2      |      94 |   49 |      74 |
| 3      | David    | M      | 0978654334 | abc123@mail.nptu.edu.tw  | 2019-01-30 16:25:31 |    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;

- 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,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 | 2019-01-30 16:24:53 |    46 | 1      |      65 |   70 |      55 |
| 2      | Cindy    | F       | NULL       | cindy900@mail.nptu.edu.tw | 2019-01-30 16:25:17 |    94 | 2      |      94 |   49 |      74 |
| 3      | David    | M       | 0978654334 | abc123@mail.nptu.edu.tw  | 2019-01-30 16:25:31 |    81 | 3      |      83 |   58 |      76 |
+--------+----------+--------+-----------+-------------------------+---------------------+-------+--------+---------+------+---------+
3 rows in set (0.00 sec)
```

Mysql/MariaDB加解密函數

- 雙向：能用key加密後也用key解回明文的，如：DES、AES。
- 單向：只能加密不能解密。這種方法多半是使用映射法產生密文，如： md5 和 sha。

雙向加密

- ENCODE(str, pass_str), DECODE(crpty_str, pass_str)
  - 加密解密字串。該函數有兩個參數：
    - 明文(str) 或密文(crpty_str) 的字串
    - 作為加密或解密基礎的金鑰（其中 pass_str 就是 key 值）
  - 這兩個函數使用有1個限制：儲存密文的欄位(以上例crpty_str)一定要為 BLOB，因為密文是二進制字串
  - 加密程度相對比較弱

雙向加密

- DES_ENCRYPT( ), DES_DECRYPT( )
- **資料加密標準**（Data Encryption Standard，DES）
  - 一種對稱密鑰加密區塊碼演算法
  - 非安全的加密方法，主要因為它使用的56位元的金鑰過短

雙向加密

- AES_ENCRYPT(str,key_str),
  AES_DECRYPT(crypt_str,key_str)
  - **進階加密標準**（**A**dvanced **E**ncryption **S**tandard，AES）
  - 替代原先的DES
  - 需要Linux
  - AES_ENCRYPT 加密結果最好以BLOB類型儲存

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));
```

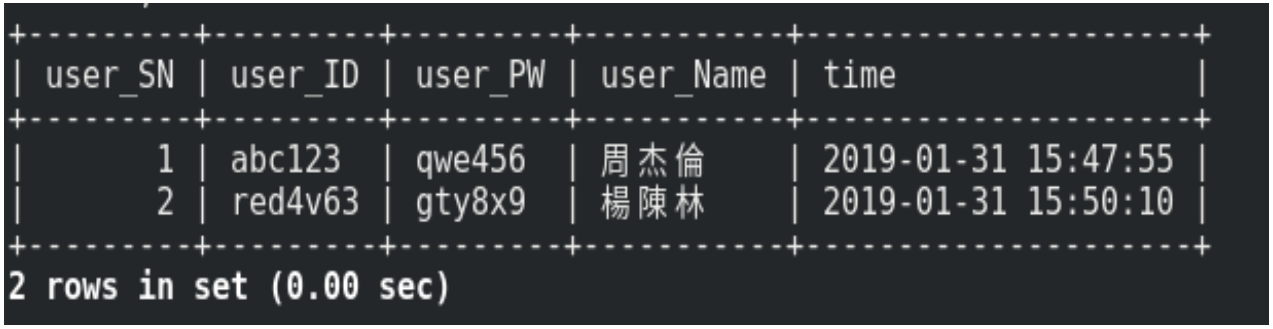
```
+-----------+----------------+------+-----+-------------------+----------------+
| Field     | Type           | Null | Key | Default           | Extra          |
+-----------+----------------+------+-----+-------------------+----------------+
| user_SN   | int(10)        | NO   | PRI | NULL              | 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('red4v63', 'gty8x9','楊陳林');



user_SN	user_ID	user_PW	user_Name	time
1	abc123	qwe456	周杰倫	2019-01-31 15:47:55
2	red4v63	gty8x9	楊陳林	2019-01-31 15:50:10

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('gty8x9','test1') where user_SN=2;

```
+---------+---------+---------+-----------+---------------------+
| user_SN | user_ID | user_PW | user_Name | time                |
+---------+---------+---------+-----------+---------------------+
|       1 | abc123  | ??^.Gb    | 周杰倫     | 2019-01-31 15:47:55 |
|       2 | red4v63 | tqq>41?   | 楊陳林     | 2019-01-31 15:50:10 |
+---------+---------+---------+-----------+---------------------+
2 rows in set (0.00 sec)

ERROR: No query specified
```

- 利用insert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     | 2019-01-31 15:47:55 |
|       2 | red4v63  | tqq>41?    | 楊陳林     | 2019-01-31 15:50:10 |
|       3 | tgh963da | t??>~?~?     | 蕭勁藤     | 2019-01-31 16:06:41 |
+---------+----------+----------+-----------+---------------------+
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    | 2019-01-31 15:47:55 | qwe456   |
|       2 | red4v63  | tqq>41?  | 楊陳林    | 2019-01-31 15:50:10 | red4v63  |
|       3 | tgh963da | t??>~?~? | 蕭勁藤    | 2019-01-31 16:06:41 | rtg964da |
+---------+----------+----------+-----------+---------------------+----------+
3 rows in set (0.00 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');

```
+---------+---------+------------------+----------+---------------------+
| user_SN | user_ID | user_PW          | user_Name | time                |
+---------+---------+------------------+----------+---------------------+
|       4 | wec5413 | ??Y?}?~?????Opc          | 林又家    | 2019-01-31 16:18:07 |
|       5 | ked963  | ?lw!Z? _}????7?       | 莫雯尉    | 2019-01-31 16:26:47 |
+---------+---------+------------------+----------+---------------------+
2 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;

user_SN	user_ID	user_PW	user_Name	time	aes_decrypt
4	wec5413	??Y?}?~?????Opc	林又家	2019-01-31 16:18:07	okg9621
5	ked963	?lw!Z? _}???7?	莫雯尉	2019-01-31 16:26:47	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');

user_SN	user_ID	user_PW	user_Name	time
6	96weq85	??????	王大福	2019-01-31 16:49:56
7	85fsw	??z?t????	張卿坊	2019-01-31 16:51:10

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;

user_SN	user_ID	user_PW	user_Name	time	des_decrypt
6	96weq85	◆◆◆◆			
yl 1;/◆R		王大福	2019-01-31 16:49:56	63gwes84	
7	85fsw	◆◆z◆t◆◆◆◆	張卿坊	2019-01-31 16:51:10	vbn3669

2 rows in set (0.00 sec)

單向加密

- ENCRYPT(,)
  - 使用UNIX crypt()系統加密字串
  - ENCRYPT( )函數接收要加密的字串和（可選的）用於加密過程的salt（一個可以唯一確定密碼的字串，就像鑰匙一樣）
  - 注意，windows上不支援

單向加密

- PASSWORD( )
  - 創建一個經過加密的密碼字串，適合於插入到MySQL的安全系統。
  - 該加密過程不可逆，和unix密碼加密過程使用不同的演算法。主要用於MySQL的認證系統。

- MD5( )
  - 一種密碼雜湊函式，可以產生出一個128 bits 雜湊值（hash value）
  - 1996年後被證實存在弱點，對於需要高度安全性的資料，專家一般建議改用其他演算法，如SHA-2。
  - 2004年證實MD5演算法無法防止碰撞（collision），因此不適用於安全性認證，如SSL公開金鑰認證或是數位簽章等用途。

- SHA( )
  - **安全雜湊演算法**（Secure Hash Algorithm）是一個密碼雜湊函式家族
  - 能計算出一個數位訊息所對應到的，長度固定的字串（又稱訊息摘要）的演算法。

SHA密碼雜湊函式家族

- SHA-0 ：
  - 1993年發布，當時稱做安全雜湊標準（Secure Hash Standard）
- SHA-1 ：
  - 1995年發布，在許多安全協定中廣為使用，包括
    - TLS
    - SSL
    - PGP
    - SSH
    - S/MIME
    - IPsec
  - 2017年宣布攻破了SHA-1

- SHA-2：
  - 2001年發布，包括
    - SHA-224
    - SHA-256
    - SHA-384
    - SHA-512
    - SHA-512/224
    - SHA-512/256。
  - 至今尚未出現對SHA-2有效的攻擊
- SHA-3:
  - 與之前SHA-1, SHA-2不同的演算法
  - 可替換的加密雜湊演算法

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');

```
+---------+----------+----------------------------------+-----------+---------------------+
| user_SN | user_ID  | user_PW                          | user_Name | time                |
+---------+----------+----------------------------------+-----------+---------------------+
|       8 | cef86314 | 47d923619176583b13ae7c92d37cf553 | 周興哲    | 2019-01-31 17:16:44 |
+---------+----------+----------------------------------+-----------+---------------------+
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)

values('asd567',password('qwdf234'),'茄子蛋');

```
 user_SN | user_ID | user_PW                                   | user_Name | time                |
---------+---------+-------------------------------------------+-----------+---------------------+
       9 | asd567  | *5D2E4A16670726F1EE88E8345189BD9C712DA478 | 茄子蛋    | 2019-01-31 17:28:24 |
---------+---------+-------------------------------------------+-----------+---------------------+
```

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');

```
+---------+---------+---------------+-----------+---------------------+
| user_SN | user_ID | user_PW       | user_Name | time                |
+---------+---------+---------------+-----------+---------------------+
|      10 | olm84k  | tenWCq/ayHJZc | 蔡一霖    | 2019-01-31 17:44:43 |
+---------+---------+---------------+-----------+---------------------+
1 row in set (0.00 sec)
```

一樣使用count比對加密後的密碼

select count(*) from Userinfo where user_SN=12 and

user_PW=encrypt('yui8413','testencrypt');

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)
 values('85osk9',sha('854ewwe'),'張匯妹');
```

```
+---------+---------+------------------------------------------+-----------+---------------------+
| user_SN | user_ID | user_PW                                  | user_Name | time                |
+---------+---------+------------------------------------------+-----------+---------------------+
|      11 | 85osk9  | fe344d892d812fdcdb5a8488087a7a79aca9a280 | 張匯妹    | 2019-01-31 17:59:13 |
+---------+---------+------------------------------------------+-----------+---------------------+
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)
```

隱碼攻擊

- SELECT * FROM user WHERE email = '$email' AND password = '$password’
- 我們輸入分別在email與password中填入：
  - ' or 1 = 1-- (最後要有空白)
- 與
  - (隨意值)

- 結果：
  - SELECT * FROM user WHERE email = ' ' or 1 = 1--' AND password = '(隨意值)'
- 以上查詢語句會等於
  - SELECT * FROM user
- 這樣就登入成功了！

- 為什麼會變成這樣呢？
- 我們先看看email的輸入：
  - ' or 1 = 1-- (最後要有空白)
- 在SQL語法中兩個單引號會等於一個單引號，前面的單引號是為了取消email的輸入格。
- or 1 = 1會讓where的值等於1(ture)。

- -- 在SQL中是所謂的註解(最後要有空白才是完整的註解)，透過註解掉後面的條件來只執行' or 1 = 1-- 的條件。

練習

- >select * from Userinfo where user_SN = '' or 1 = 1-- ' and user_PW ='123’;
  >;