快取指令

資料庫備份還原

備份情境	指令範例	說明
匯出單一資料庫	`mysqldump -u user -p dbname > backup.sql`	最常見的備份方式，匯出整個資料庫。
匯出特定資料表	`mysqldump -u user -p dbname tablename > table.sql`	只匯出某一張表格。
匯出多個資料庫	`mysqldump -u user -p --databases db1 db2 > multi.sql`	同時匯出多個指定資料庫。
匯出所有資料庫	`mysqldump -u user -p --all-databases > all.sql`	匯出整個 MariaDB 伺服器所有資料庫。
匯出含儲存程序/觸發器	`mysqldump -u user -p --routines --triggers dbname > full.sql`	包含 Stored Procedures 與 Triggers。
匯出大型資料庫 (避免鎖表)	`mysqldump -u user -p --single-transaction dbname > backup.sql`	適合 InnoDB，大型資料庫匯出時避免鎖表。
匯出含事件 (Events)	`mysqldump -u user -p --events dbname > events.sql`	匯出排程事件。

參數:

- `-u`：指定使用者帳號
- `-p`：輸入密碼（執行後會提示輸入）
- `資料庫名稱`: 要匯出的資料庫
- `backup.sql`：匯出的檔案名稱
- `--single-transaction`：匯出時避免鎖表，適合大型資料庫。
- `--routines`：包含儲存程序 (Stored Procedures)。
- `--triggers`：包含觸發器 (Triggers)。
- `--events`：包含事件 (Events)。
- [`mysqldump` [`--databases` [`--all-databases`。
- [`--single-transaction`。
- 若有儲存程序、觸發器、事件，記得加上對應參數。

資料庫形式

字串類

char(size)

最多255字元。
varchar(size)
固定字元。
tinytext(size)
最多255字元。
binary(size)
最大長度為255個位元組。
text(size)
最多可存放65535個字元。
mediumtext(size)
最多能放入16777215個字元。
longtext(size)
longtext的最大容量高達4GB，也就是4,294,967,295個字元。

數質類

bit

只能存0與1
tinyint
\-128到127，無正負號（也就是非負整數）的表達範圍從0到255。
smallint
從-32768到32767，無號值範圍從0到65535。
mediumint
從-8388608到8388607，無符號值範圍從0到16777215。
int(integer)
\-2147483648到2147483647，無符號值範圍從0到4294967295。
bigint
從-9223372036854775808到9223372036854775807，無符號值範圍從0到18446744073709551615。 float
單精度浮點數，在記憶體中需4個位元組的空間。
double
雙精度浮點數，它相較於float多了一倍的表達範圍，需8個位元組的儲存空間。
Signed
欄位有負數。
Unsigned
欄位只有正數。
Length
與 ZeroFill 一起使用，表示資料的預設長度。
ZeroFill
欄位前方會自動補 0，補到資料等於預設長度，如果指定 ZeroFill，會自動設定 Unsigned。 AUTO_INCREMENT

欄位不需指定數值，系統會自動遞增，通常是用來產生唯一的識別碼。

日期時間資料型別

date

從"1000-01-01"到"9999-12-31"，顯示格式為：'yyyy-mm-dd'。
datetime
從"1000-01-01 00:00:00"到"9999-12-31 23:59:59"，顯示格式為：'yyyy-mm-dd hh:mm:ss'。 timestamp(m)
從'1970-01-01 00:00:01' utc到'2038-01-19 03:14:07' utc，顯示格式為：'yyyy-mm-dd hh:mm:ss'。 time
從'-838：59：59'到'838：59：59'，顯示格式為：'hh:mm:ss'。
year[(2/4)]
有2位或4位數的差別。預設4位數

大物件(LOB)資料型別

tinyblob

最大為255個位元組。
blob(size)
最大為65535位元組。
mediumblob
最大為16777215位元組。
longtext
最大尺寸4GB或4,294,967,295個字元。

其他型別

INET6
IPv6 MAC
JSON
額外設定 json_valid 來驗證插入的值是否為有效的 JSON 格式。 ENUM
清單最多支援 65,535 個不同的值。
SET
清單最多支援 64 個不同的值。

create database 資料庫名稱; #建立資料庫
show databases; #顯示資料庫

SELECT VERSION(); #檢視版本
SELECT USER(); #檢視使用者
SHOW

use 資料庫名稱;　　#使用資料庫

create table table名稱 (
->
->
->
->);

```
MariaDB [(none)]> use demo;
Database changed
MariaDB [demo]> create table result (
    ->        event_id MEDIUMINT NOT NULL AUTO_INCREMENT,
    ->        status varchar(30),
    ->        date datetime,
    ->        primary key (event_id)
    ->  );
```

drop database 資料庫名稱;　　#刪除資料庫
drop table 表格名稱;
alter table表格名稱 drop column 查詢表名稱;

#也可以使用add 跟一次做多個查詢表的處理

show columns from 資料表 from 資料庫;　　#查詢表格初始狀況

備註: 也可以使用以下方式

describe 資料表;
desc 資料表;

select user,host from mysql.user;　　#查詢登入者與登入主機位置

rename user 'root'@'127.0.0.1' to 'imaghost'@'127.0.0.1';　#變更使用者名稱

create user 'username'@'hostname' identified by '密碼';　　#新增使用者

備註: hostname更改成%代表任何主機

```
grant all on 資料庫名稱.* to 'username'@'localhost';
or
grant SELECT,INSERT,UPDATE,DELETE on 資料庫名稱.* to 'username'@'hostname';
```

將資料庫權限給該使用者

```
show grants for 'user'@'hostname';   #顯示使用者權限
```

```
show grants;          #顯示使用者權限
```

```
revoke all privileges on 資料庫名稱.table from 'username'@'hostname';
```

移除改使用者權限

```
revoke all privileges,grant option from 'username'@'hostname';
```

移除所有使用者權限

```
drop user 'username'@'hostname';      #移除使用者
```

```
mysql_secure_installation #安全設定
```

select 查詢

撈取資料表中所有欄位及資料:

```
select * from `資料表名稱`;
select * from `news`;
```

指定撈取部分欄位:

```
select `欄位一`, `欄位二` from `資料表名稱`;
select `id`, `name` from `news`;
```

where 條件

撈取欄位一等於 "1" 的資料

```
select * from `資料表名稱` where `欄位一` = 1;
select * from `news` where `id` = 1;
```

撈取欄位一等於"1"，且欄位二大於"10"的資料

```
select * from `□□□□□` where `□□□` = 1 and `□□□` > 10;
select * from `news` where `id` = 1 and `pageviews` > 10;
```

where in □□ □□

□□□□□□□ "1" □ "2" □ "3" □□□

```
select * from `□□□□□` where `□□□` in (1, 2, 3);
select * from `news` where `id` in (1, 2, 3);
```

where is null □□ □□ null

□□□□□□ "null" □□□

```
select * from `□□□□□` where `□□□` is null;
select * from `news` where `id` is null;
```

where is not null □□ □□□ null

□□□□□□□ "null" □□□

```
select * from `□□□□□` where `□□□` is not null;
select * from `news` where `id` is not null;
```

order by □□

□□□□□□□□□□□□□

```
select * from `□□□□□` order by `□□□` asc;
select * from `news` order by `id` asc;
```

□□□□□□□□□□□□□

```
select * from `□□□□□` order by `□□□` desc;
select * from `news` order by `id` desc;
```

□□□□□□□□□□□□□

```
select * from `□□□□□` order by binary(CONVERT(`□□□` using big5))
select * from news order by binary(CONVERT(`name` using big5))
```

limit □□□□

取得五筆資料

```
SELECT * FROM `資料表名稱` LIMIT 5
SELECT * FROM `news` LIMIT 5
```

取得五筆資料, 並由第十筆開始取得(得到排序中第10\~15筆)

```
SELECT * FROM `資料表名稱` LIMIT 5,10
SELECT * FROM `news` LIMIT 5,10
```

group 群組

取得資料並將欄位一相同的資料合併為一筆

```
SELECT * FROM `資料表名稱` GROUP BY `欄位一`;
SELECT * FROM `news` GROUP BY `release_date`;
```

計算相同欄位一相同資料的欄位加總(取得每天新聞各有幾筆)

```
SELECT COUNT(*) FROM `資料表名稱` GROUP BY `欄位一`;
SELECT COUNT(*) FROM `news` GROUP BY `release_date`;
```

insert 插入資料

插入一筆新資料

```
insert into `資料表名稱`(`欄位一`,`欄位二`) VALUES ('欄位一資料', '欄位二資料');
insert into `news`(`title`,`release_date`) VALUES ('焦點新聞快報', '1990-05-09');
```

ALTER TABLE 插入一個新欄位

```
ALTER TABLE `資料表名稱` ADD `新增欄位名稱` 新增欄位型態 新增欄位預設值 COMMENT '新增欄位說明';
ALTER TABLE `news` ADD deleted_at TIMESTAMP NULL COMMENT '刪除時間';
```

update 更新資料

```
UPDATE `資料表名稱` SET `欄位一` = '欄位一資料', `欄位二` = '欄位二資料';
UPDATE `news` SET `title` = '焦點新聞快報!', `release_date` = '2020-02-02';
```

5.更新使用子查询的值:

SQL 语句通过子查询计算每个 'Premium' 类型客户的总购买金额，并将该值更新到 total_purchases 列中。

```
UPDATE customers
SET total_purchases = (
    SELECT SUM(amount)
    FROM orders
    WHERE orders.customer_id = customers.customer_id
)
WHERE customer_type = 'Premium';
```

□□□□

```
update '□□□' pwd=md5(pwd);
```

```
1 | UPDATE `testmd5` SET pwd=MD5(pwd);
```

运行后刷新得到结果

1 信息 | 2 表数据 | 3 信息

id	name	pwd
1	张三	1f32aa4c9a1d2ea010adcf2348166a04
4	小陈	25d55ad283aa400af464c76d713c07ad
2	李四	e10adc3949ba59abbe56e057f20f883e
3	王五	fcea920f7412b5da7be0cf42b8c93759
*	(NULL)	(NULL)

```
1 | UPDATE `testmd5` SET pwd=MD5(pwd);
```

运行后刷新得到结果

1 信息 | 2 表数据 | 3 信息

id	name	pwd
1	张三	1f32aa4c9a1d2ea010adcf2348166a04
4	小陈	25d55ad283aa400af464c76d713c07ad
2	李四	e10adc3949ba59abbe56e057f20f883e
3	王五	fcea920f7412b5da7be0cf42b8c93759
*	(NULL)	(NULL)

delete □□□□

□□□□□□□□□□ "1" □□□□

```
delete from `□□□□` WHERE `□□` = 1;
delete from `news` WHERE `id` = 1;
```

truncate □□

□□□□□□□□□□□

```
truncate table `□□□□□`;
truncate table `news`;
```

□□□□md5

```
md5('□□□□')
```

Revision #2
Created 18 September 2024 14:00:15 by Ron
Updated 17 March 2026 03:37:48 by Ron