PostgreSQL

一、前言

1.切換數據庫(作為系統管理員)

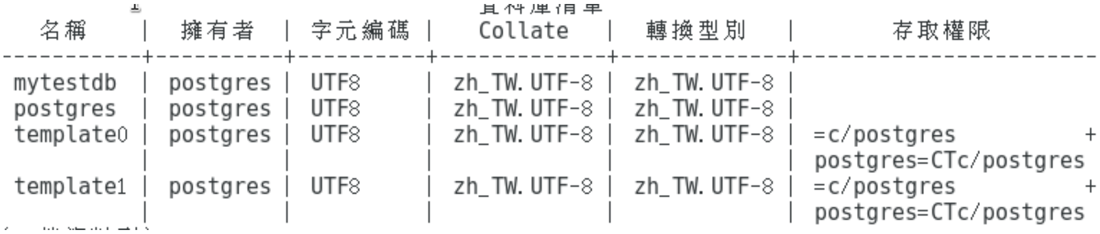
sudo su postgres

2.版本

psql –version

3.數據庫列表

psql –l

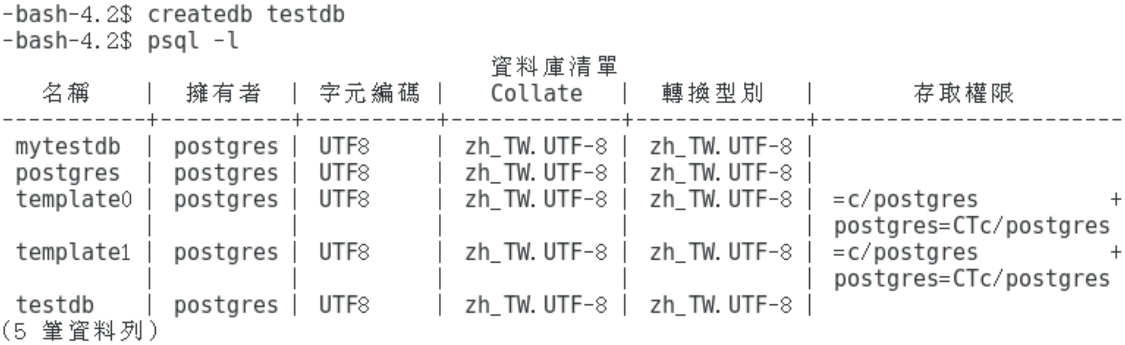


```
                                   資料庫清單
   名稱     |  擁有者  | 字元編碼 |   Collate    |   轉換型別   |       存取權限
-----------+----------+----------+-------------+-------------+-----------------------
 mytestdb  | postgres | UTF8     | zh_TW.UTF-8 | zh_TW.UTF-8 |
 postgres  | postgres | UTF8     | zh_TW.UTF-8 | zh_TW.UTF-8 |
 template0 | postgres | UTF8     | zh_TW.UTF-8 | zh_TW.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
 template1 | postgres | UTF8     | zh_TW.UTF-8 | zh_TW.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
```

4.建數據庫

createdb 數據庫名稱

e.g createdb testdb

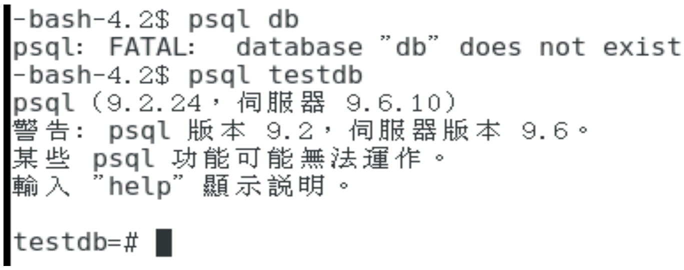


```
-bash-4.2$ createdb testdb
-bash-4.2$ psql -l
                                   資料庫清單
   名稱     |  擁有者  | 字元編碼 |   Collate    |   轉換型別   |       存取權限
-----------+----------+----------+-------------+-------------+-----------------------
 mytestdb  | postgres | UTF8     | zh_TW.UTF-8 | zh_TW.UTF-8 |
 postgres  | postgres | UTF8     | zh_TW.UTF-8 | zh_TW.UTF-8 |
 template0 | postgres | UTF8     | zh_TW.UTF-8 | zh_TW.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
 template1 | postgres | UTF8     | zh_TW.UTF-8 | zh_TW.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
 testdb    | postgres | UTF8     | zh_TW.UTF-8 | zh_TW.UTF-8 |
(5 筆資料列)
```

5.進入 psql 命令模式，並進入數據庫

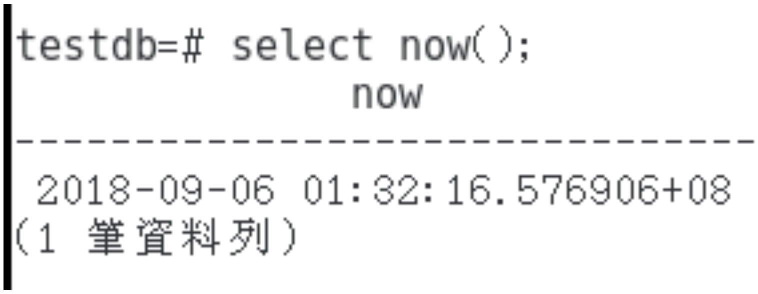
psql testdb

會有以下畫面



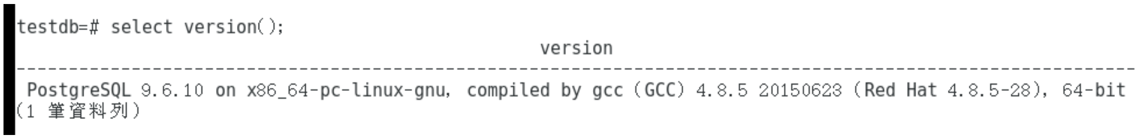
```
-bash-4.2$ psql db
psql: FATAL:  database "db" does not exist
-bash-4.2$ psql testdb
psql (9.2.24，伺服器 9.6.10)
警告: psql 版本 9.2，伺服器版本 9.6。
某些 psql 功能可能無法運作。
輸入 "help" 顯示說明。

testdb=# █
```

6. now



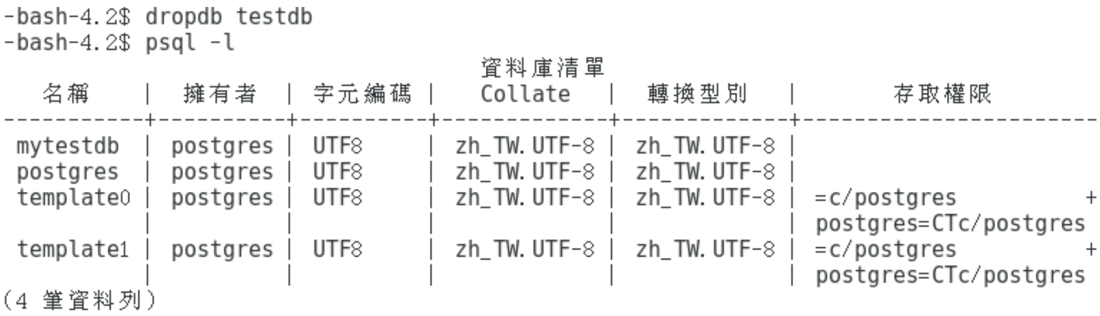
```
testdb=# select now();
              now
-------------------------------
 2018-09-06 01:32:16.576906+08
(1 筆資料列)
```

7.version



```
testdb=# select version();
                                                  version
---------------------------------------------------------------------------------------------------------
 PostgreSQL 9.6.10 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-28), 64-bit
(1 筆資料列)
```

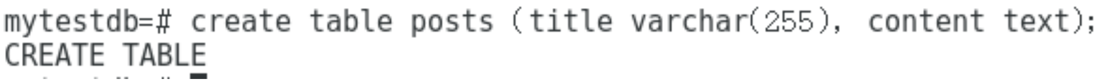
8.刪除數據庫



```
-bash-4.2$ dropdb testdb
-bash-4.2$ psql -l
                                    資料庫清單
   名稱    |  擁有者  | 字元編碼 |   Collate   |  轉換型別   |       存取權限
-----------+----------+----------+-------------+-------------+-----------------------
 mytestdb  | postgres | UTF8     | zh_TW.UTF-8 | zh_TW.UTF-8 |
 postgres  | postgres | UTF8     | zh_TW.UTF-8 | zh_TW.UTF-8 |
 template0 | postgres | UTF8     | zh_TW.UTF-8 | zh_TW.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
 template1 | postgres | UTF8     | zh_TW.UTF-8 | zh_TW.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
(4 筆資料列)
```

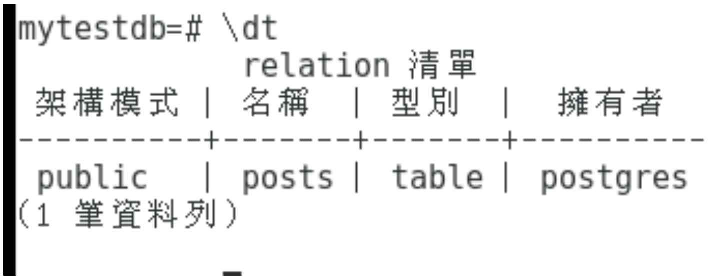
二、操作表數據

1.建表(varchar 可變長字串 255, text 大文本類型)



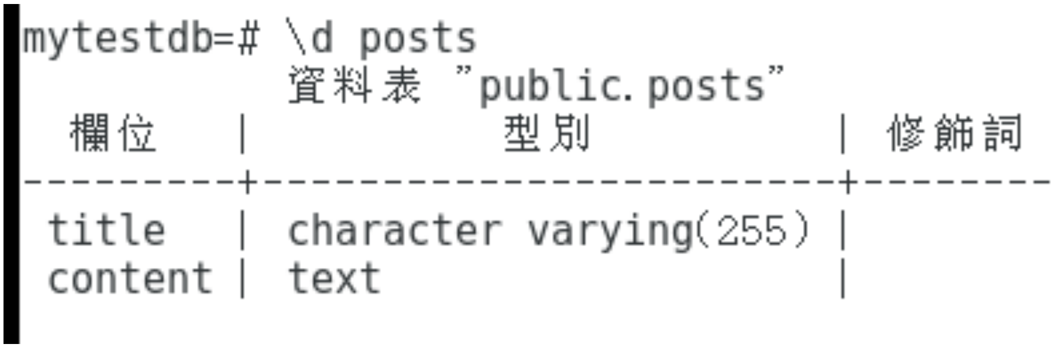
```
mytestdb=# create table posts (title varchar(255), content text);
CREATE TABLE
```

2.查看訊息： \dt



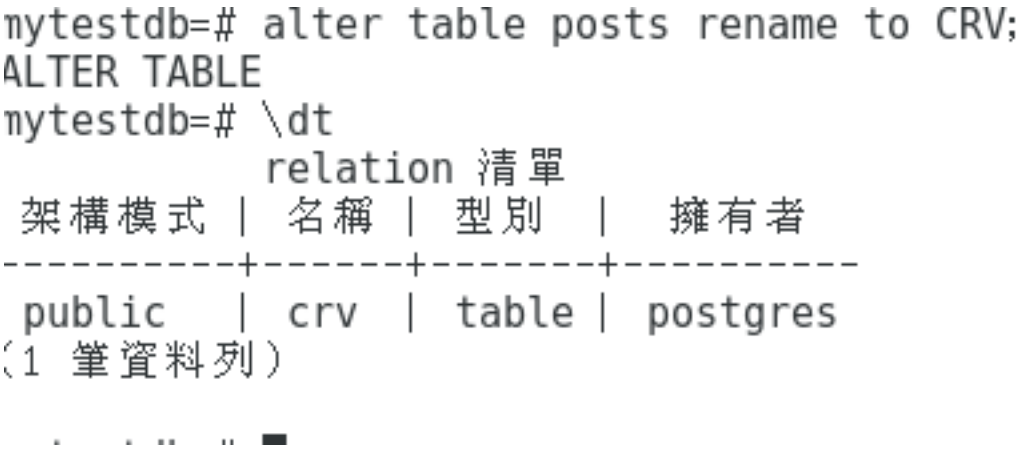
```
mytestdb=# \dt
           relation 清單
 架構模式 | 名稱  | 型別  |  擁有者
----------+-------+-------+----------
 public   | posts | table | postgres
(1 筆資料列)
```

3.獲取表的詳細內容：\d 表名



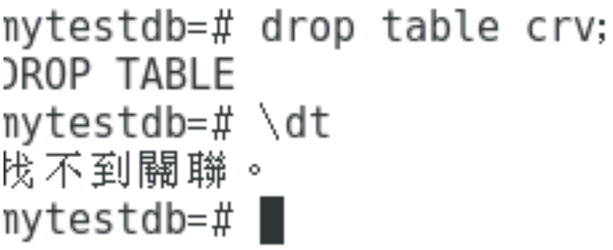
```
mytestdb=# \d posts
          資料表 "public.posts"
  欄位   |          型別          | 修飾詞
---------+------------------------+--------
 title   | character varying(255) |
 content | text                   |
```

4.變更表名



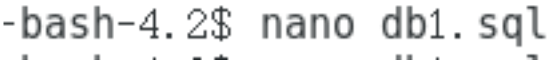
```
mytestdb=# alter table posts rename to CRV;
ALTER TABLE
mytestdb=# \dt
          relation 清單
 架構模式 | 名稱 | 型別  |  擁有者
----------+------+-------+----------
 public   | crv  | table | postgres
(1 筆資料列)
```

5.刪除表

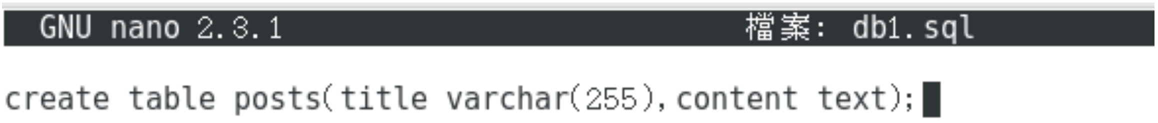


```
nytestdb=# drop table crv;
)ROP TABLE
nytestdb=# \dt
戈不到關聯。
nytestdb=# 
```

6.建數據庫 SQL(nano 方式)

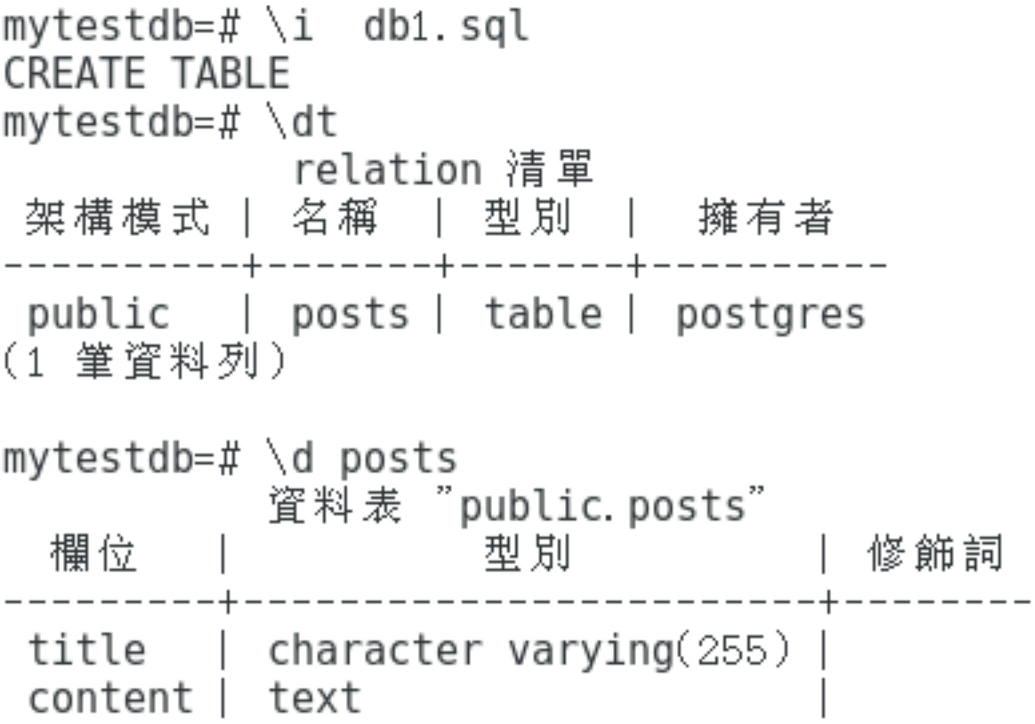


```
-bash-4.2$ nano db1.sql
```



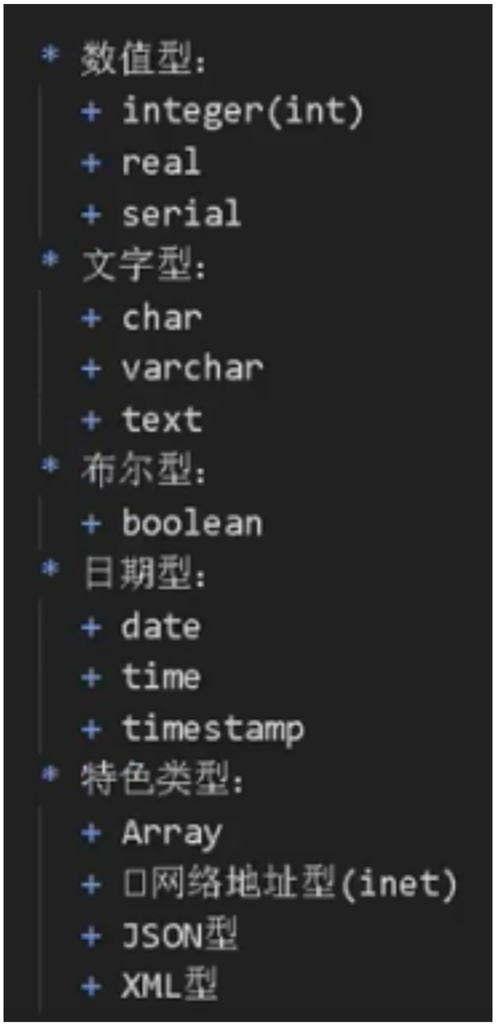
```
  GNU nano 2.3.1                           檔案: db1.sql

create table posts(title varchar(255),content text);
```

\i 引入 SQL



```
mytestdb=# \i  db1.sql
CREATE TABLE
mytestdb=# \dt
            relation 清單
 架構模式 | 名稱  | 型別  |  擁有者
----------+-------+-------+----------
 public   | posts | table | postgres
(1 筆資料列)

mytestdb=# \d posts
           資料表 "public.posts"
  欄位   |          型別          | 修飾詞
---------+------------------------+--------
 title   | character varying(255) |
 content | text                   |
```

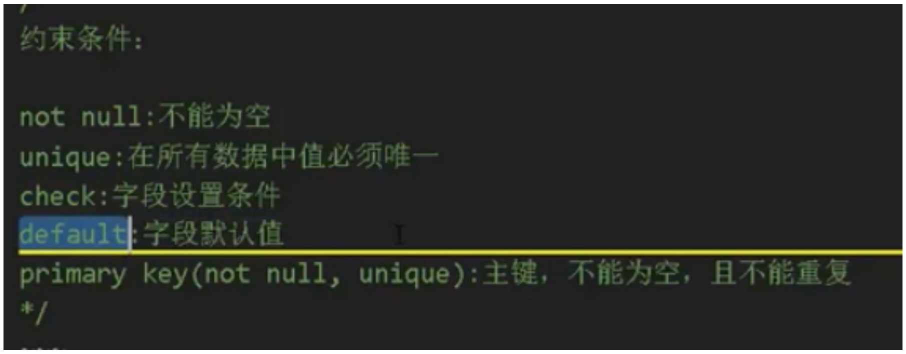
三、字段類型



* 数值型:
  + integer(int)
  + real
  + serial
* 文字型:
  + char
  + varchar
  + text
* 布尔型:
  + boolean
* 日期型:
  + date
  + time
  + timestamp
* 特色类型:
  + Array
  + 口网络地址型(inet)
  + JSON型
  + XML型

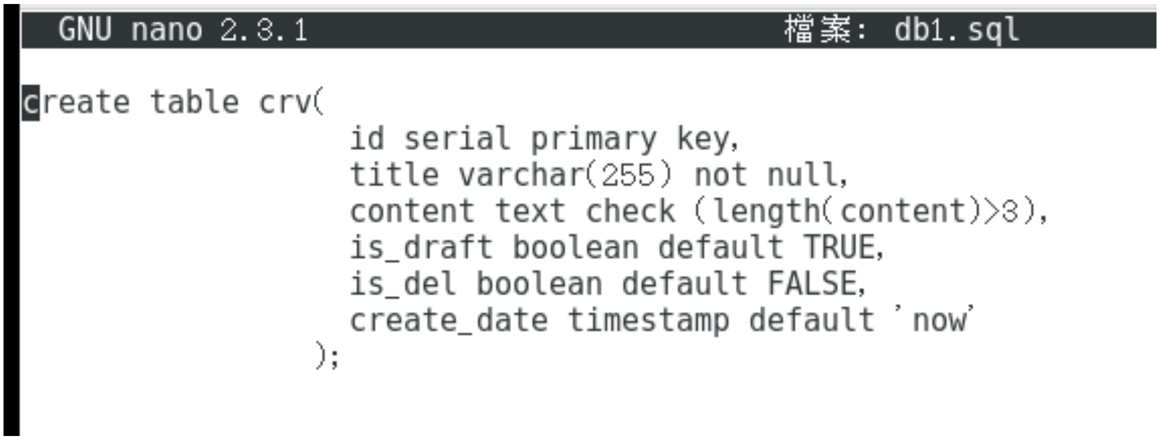
四、添加表約束

約束條件



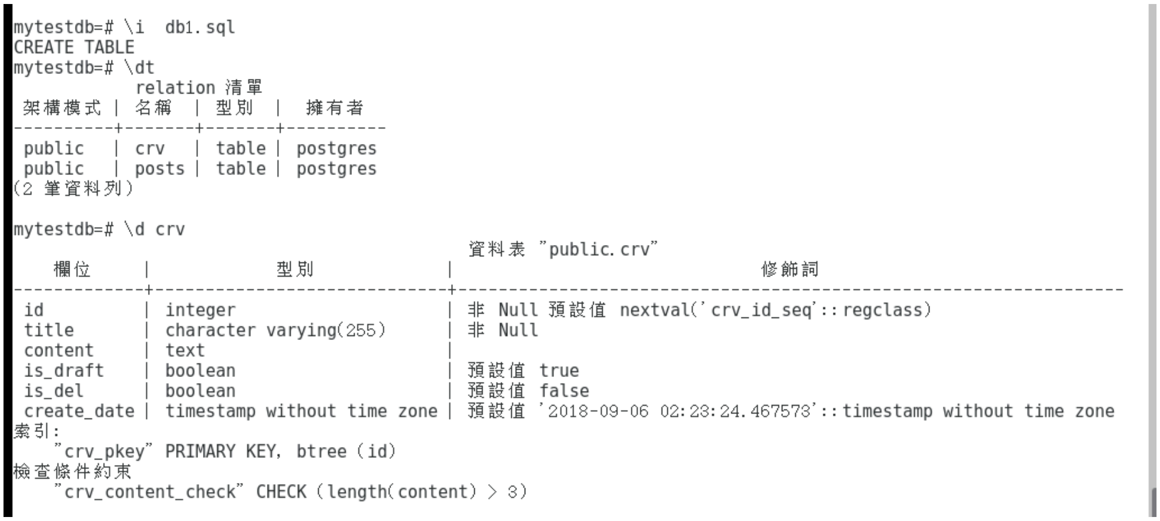
```
约束条件:

not null:不能为空
unique:在所有数据中值必须唯一
check:字段设置条件
default:字段默认值
primary key(not null, unique):主键，不能为空，且不能重复
*/
```

寫入約束條件



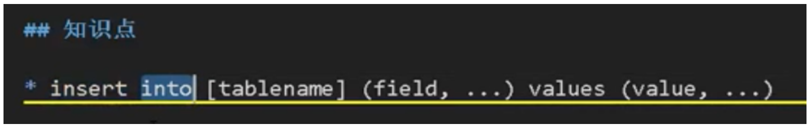
```
create table crv(
                id serial primary key,
                title varchar(255) not null,
                content text check (length(content)>3),
                is_draft boolean default TRUE,
                is_del boolean default FALSE,
                create_date timestamp default 'now'
              );
```

e.g.

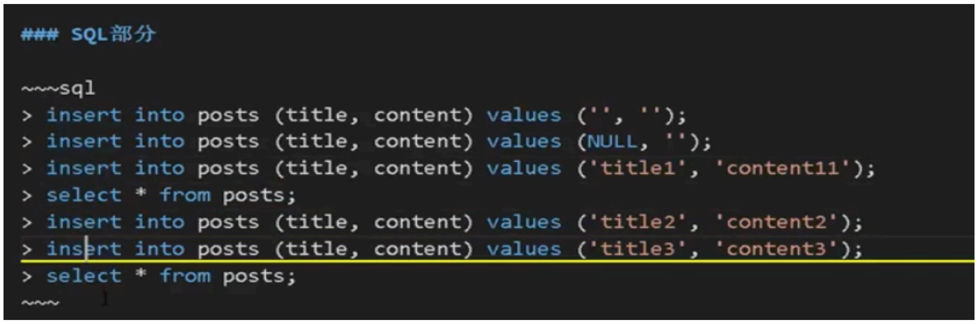


```
mytestdb=# \i  db1.sql
CREATE TABLE
mytestdb=# \dt
            relation 清單
 架構模式 | 名稱  | 型別  |  擁有者
----------+-------+-------+----------
 public   | crv   | table | postgres
 public   | posts | table | postgres
(2 筆資料列)

mytestdb=# \d crv
                                             資料表 "public.crv"
    欄位     |             型別            |                              修飾詞
-------------+-----------------------------+-------------------------------------------------------------------
 id          | integer                     | 非 Null 預設值 nextval('crv_id_seq'::regclass)
 title       | character varying(255)      | 非 Null
 content     | text                        |
 is_draft    | boolean                     | 預設值 true
 is_del      | boolean                     | 預設值 false
 create_date | timestamp without time zone | 預設值 '2018-09-06 02:23:24.467573'::timestamp without time zone
索引:
    "crv_pkey" PRIMARY KEY, btree (id)
檢查條件約束
    "crv_content_check" CHECK (length(content) > 3)
```

五、INSERT 語句

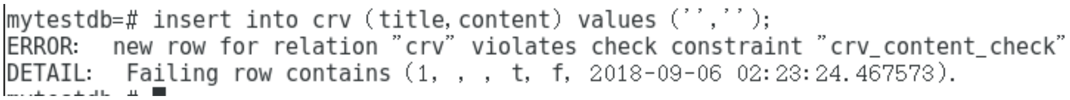


```
## 知识点

* insert into [tablename] (field, ...) values (value, ...)
```



```
### SQL部分

~~~sql
> insert into posts (title, content) values ('', '');
> insert into posts (title, content) values (NULL, '');
> insert into posts (title, content) values ('title1', 'content11');
> select * from posts;
> insert into posts (title, content) values ('title2', 'content2');
> insert into posts (title, content) values ('title3', 'content3');
> select * from posts;
~~~
```

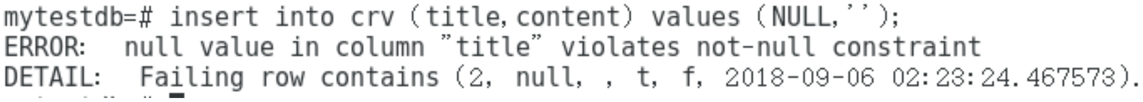
1.title,content 都插入空字串



```
mytestdb=# insert into crv（title,content）values（’’,’’）;
ERROR:  new row for relation ”crv” violates check constraint ”crv_content_check”
DETAIL:  Failing row contains（1, , , t, f, 2018-09-06 02:23:24.467573).
```

crv_content_check 有 error， 空字串‘’是有值，但小於 3

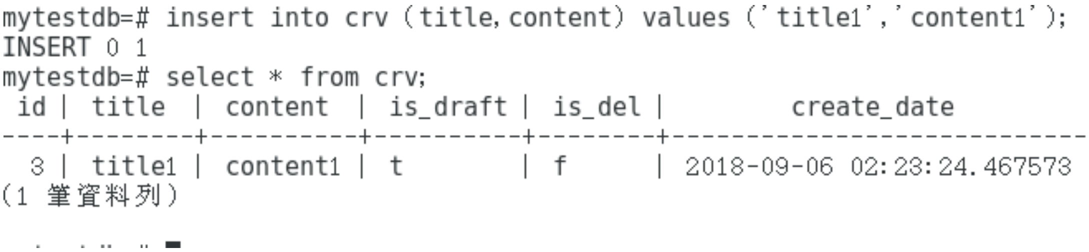
2.title 插入 NULL，回覆 error



```
mytestdb=# insert into crv（title,content）values（NULL,’’）;
ERROR:  null value in column ”title” violates not-null constraint
DETAIL:  Failing row contains（2, null, , t, f, 2018-09-06 02:23:24.467573).
```

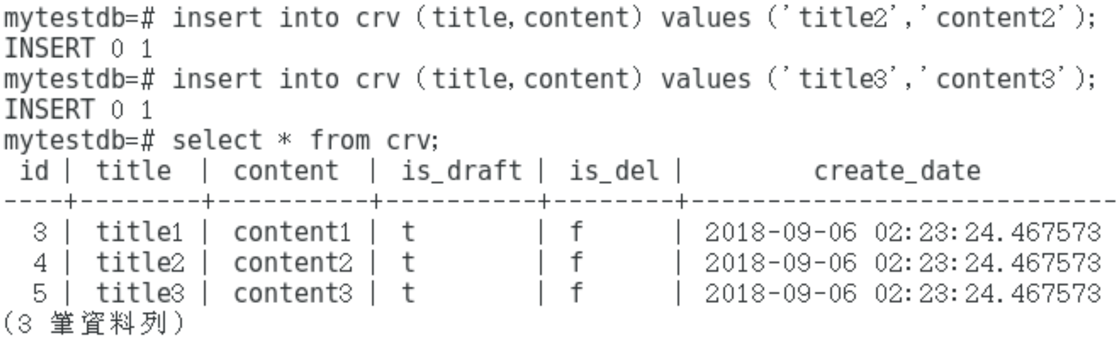
3.插入成功的紀錄並查看

id=3(當初使用 serial)代表做了 3 次，前 2 次失敗



```
mytestdb=# insert into crv（title,content）values（’title1’,’content1’）;
INSERT 0 1
mytestdb=# select * from crv;
 id | title  | content  | is_draft | is_del |        create_date
----+--------+----------+----------+--------+----------------------------
  3 | title1 | content1 | t        | f      | 2018-09-06 02:23:24.467573
（1 筆資料列）
```

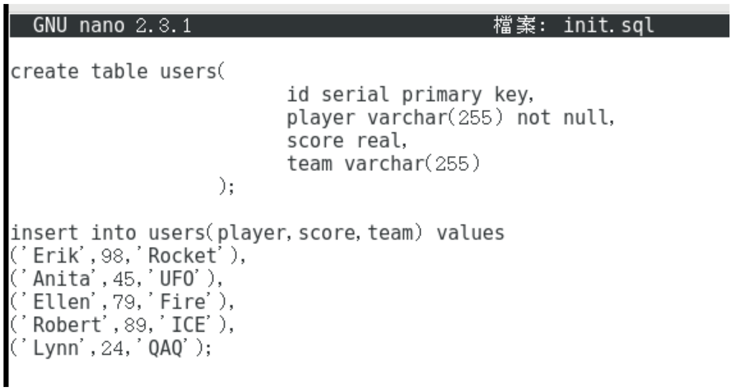
繼續插入資料



```
mytestdb=# insert into crv（title,content）values（’title2’,’content2’）;
INSERT 0 1
mytestdb=# insert into crv（title,content）values（’title3’,’content3’）;
INSERT 0 1
mytestdb=# select * from crv;
 id | title  | content  | is_draft | is_del |        create_date
----+--------+----------+----------+--------+----------------------------
  3 | title1 | content1 | t        | f      | 2018-09-06 02:23:24.467573
  4 | title2 | content2 | t        | f      | 2018-09-06 02:23:24.467573
  5 | title3 | content3 | t        | f      | 2018-09-06 02:23:24.467573
（3 筆資料列）
```

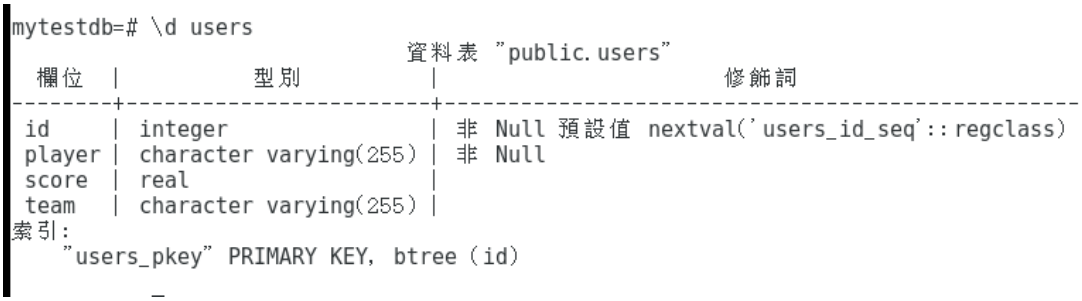
六、SELECT 語句

1.建立數據庫



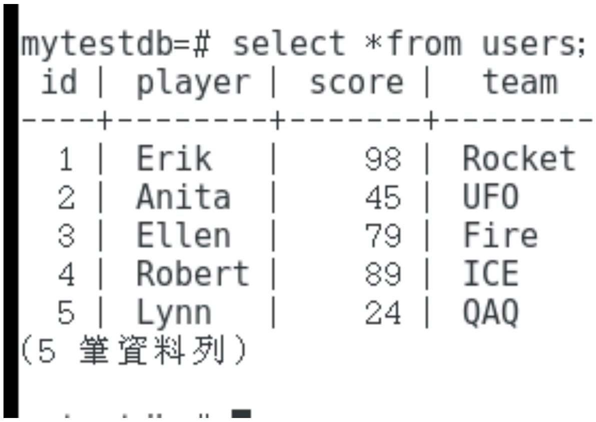
```
  GNU nano 2.3.1                          檔案: init.sql

create table users(
                        id serial primary key,
                        player varchar(255) not null,
                        score real,
                        team varchar(255)
                  );

insert into users(player,score,team) values
(’Erik’,98,’Rocket’),
(’Anita’,45,’UFO’),
(’Ellen’,79,’Fire’),
(’Robert’,89,’ICE’),
(’Lynn’,24,’QAQ’);
```

2.查看訊息



```
mytestdb=# \d users
                                  資料表 ”public.users”
  欄位  |          型別          |                    修飾詞
--------+------------------------+---------------------------------------------
 id     | integer                | 非 Null 預設值 nextval(’users_id_seq’::regclass)
 player | character varying(255) | 非 Null
 score  | real                   |
 team   | character varying(255) |
索引:
    ”users_pkey” PRIMARY KEY, btree (id)
```

3.查看資料庫

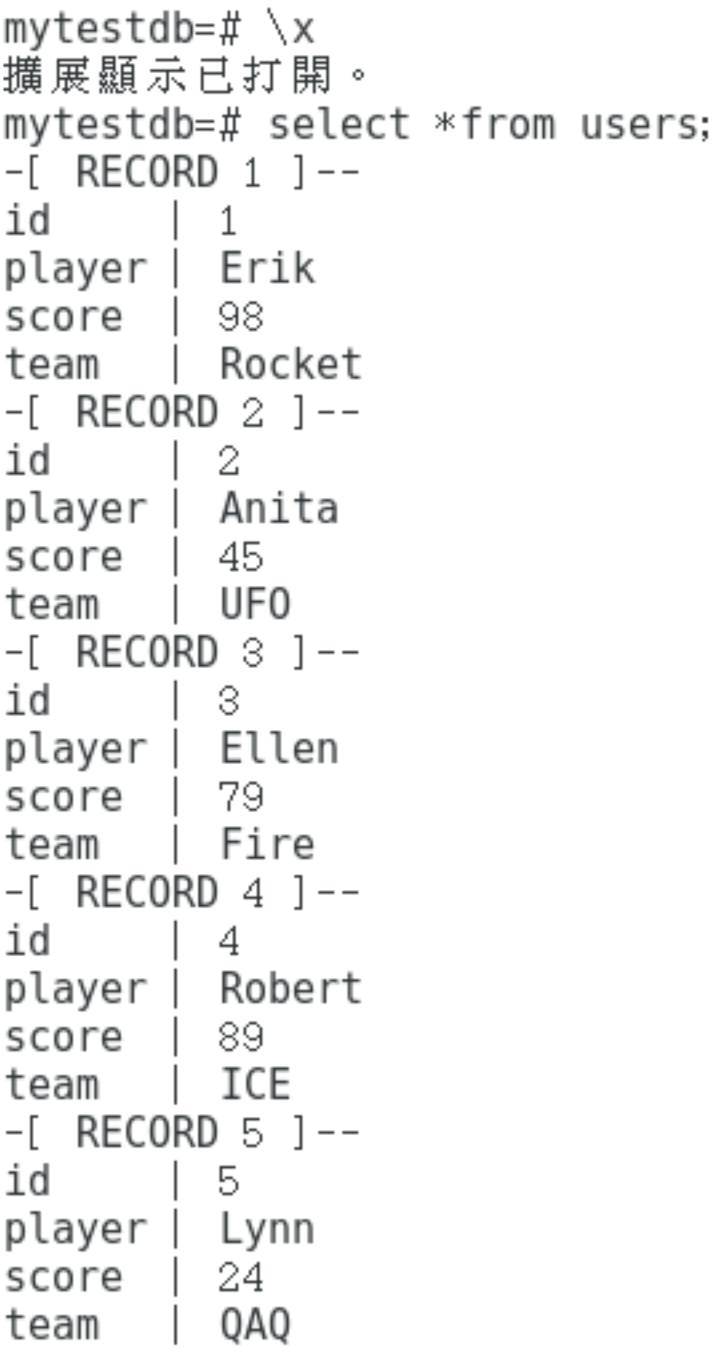


```
mytestdb=# select *from users;
 id | player | score |  team
----+--------+-------+--------
  1 | Erik   |    98 | Rocket
  2 | Anita  |    45 | UFO
  3 | Ellen  |    79 | Fire
  4 | Robert |    89 | ICE
  5 | Lynn   |    24 | QAQ
(5 筆資料列)
```

4.若資料表過大可使用以下方式檢視

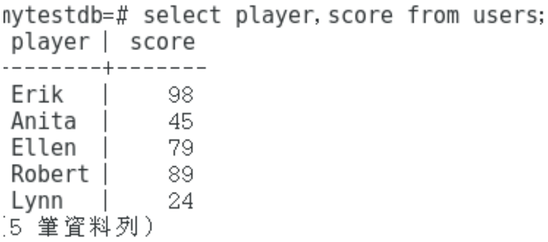
\x

再打一次就會關閉



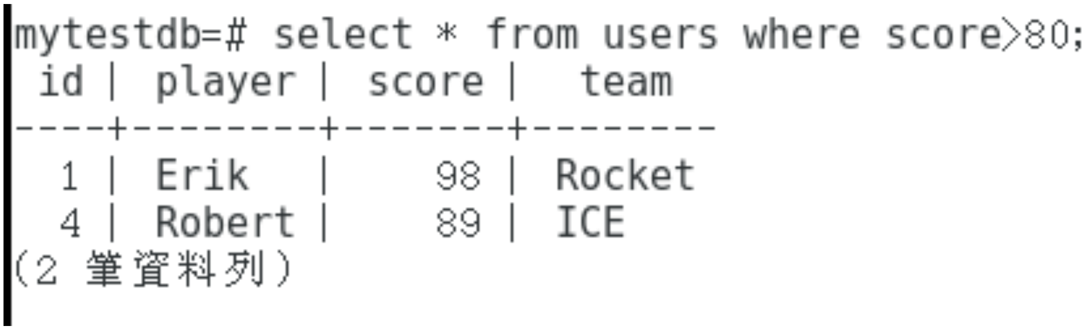
```
mytestdb=# \x
擴展顯示已打開。
mytestdb=# select *from users;
-[ RECORD 1 ]--
id     | 1
player | Erik
score  | 98
team   | Rocket
-[ RECORD 2 ]--
id     | 2
player | Anita
score  | 45
team   | UFO
-[ RECORD 3 ]--
id     | 3
player | Ellen
score  | 79
team   | Fire
-[ RECORD 4 ]--
id     | 4
player | Robert
score  | 89
team   | ICE
-[ RECORD 5 ]--
id     | 5
player | Lynn
score  | 24
team   | QAQ
```

5. select 某些

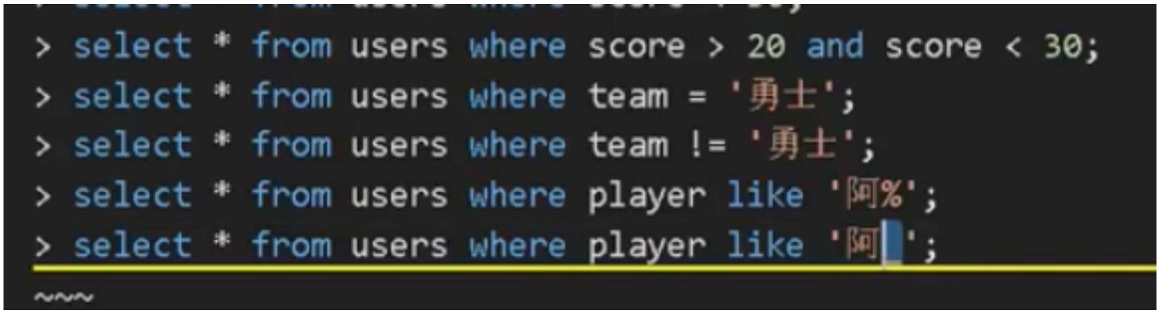


```
nytestdb=# select player,score from users;
 player | score
--------+-------
 Erik   |    98
 Anita  |    45
 Ellen  |    79
 Robert |    89
 Lynn   |    24
(5 筆資料列)
```

七、WHERE 語句



```
mytestdb=# select * from users where score>80;
 id | player | score |  team
----+--------+-------+--------
  1 | Erik   |    98 | Rocket
  4 | Robert |    89 | ICE
(2 筆資料列)
```



```
> select * from users where score > 20 and score < 30;
> select * from users where team = '勇士';
> select * from users where team != '勇士';
> select * from users where player like '阿%';
> select * from users where player like '阿_';
```

八、數據抽出選項

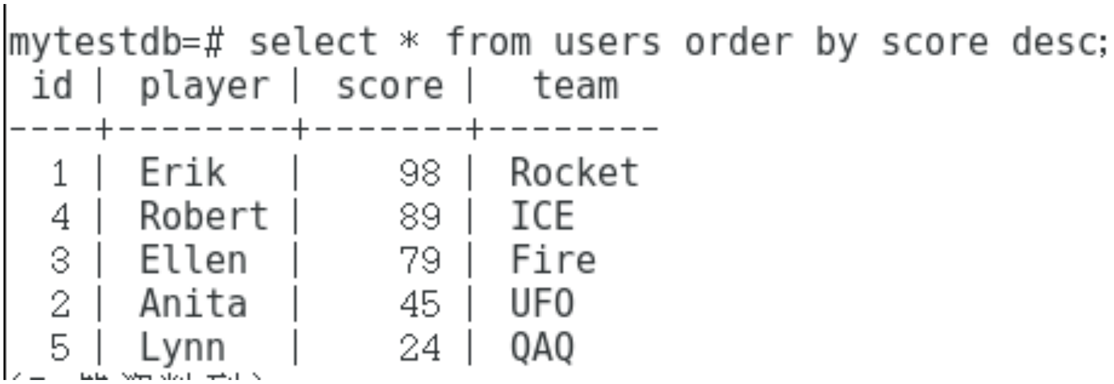
order by

limit

offset

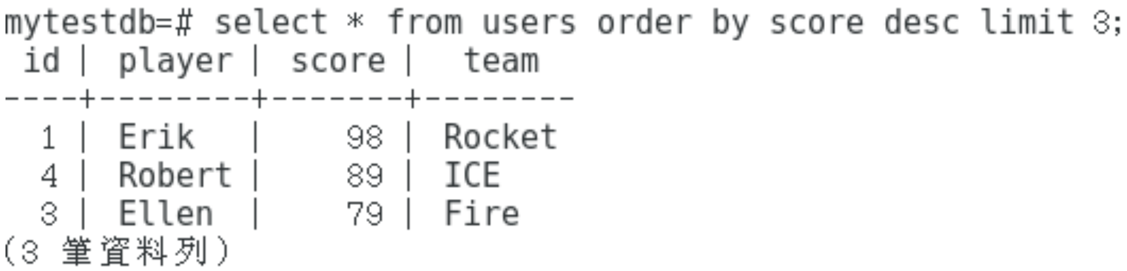
Order by 概念，並加上取出滿足條件的前幾個

score 降冪排序



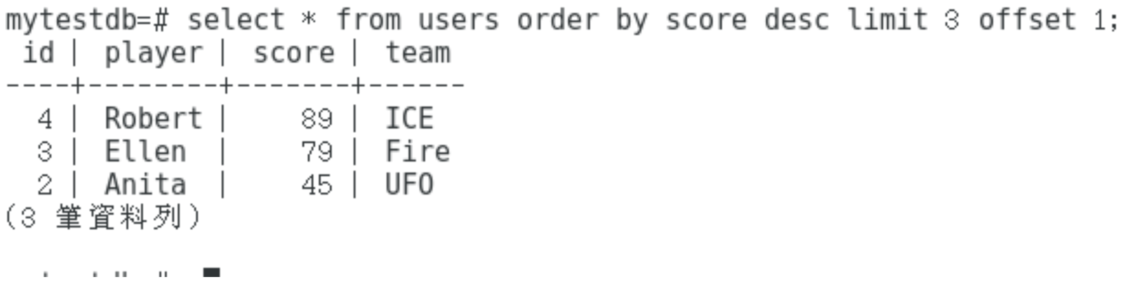
```
mytestdb=# select * from users order by score desc;
 id | player | score |  team
----+--------+-------+--------
  1 | Erik   |    98 | Rocket
  4 | Robert |    89 | ICE
  3 | Ellen  |    79 | Fire
  2 | Anita  |    45 | UFO
  5 | Lynn   |    24 | QAQ
```

有 limit，只看前 3 個



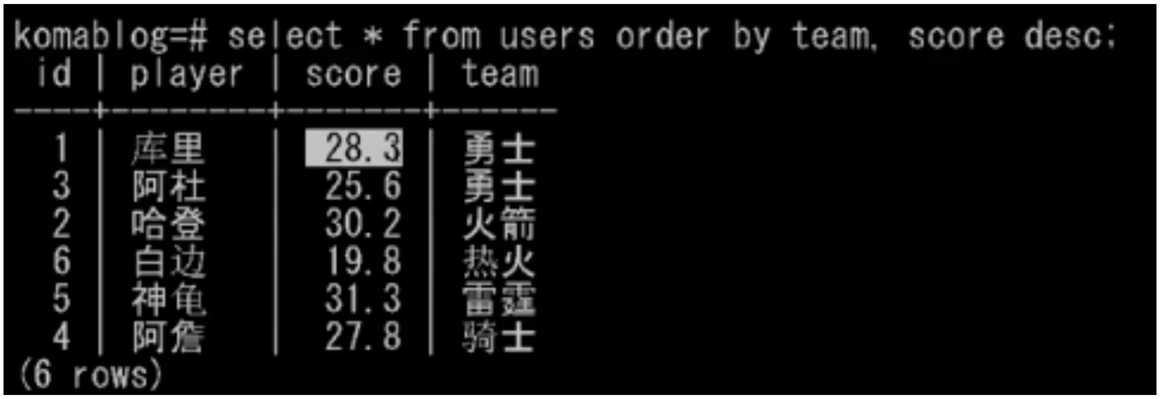
```
mytestdb=# select * from users order by score desc limit 3;
 id | player | score |  team
----+--------+-------+--------
  1 | Erik   |    98 | Rocket
  4 | Robert |    89 | ICE
  3 | Ellen  |    79 | Fire
(3 筆資料列)
```

有 limit 及有 offset，這樣會第一名會被踢除，從第二名到第四名



```
mytestdb=# select * from users order by score desc limit 3 offset 1;
 id | player | score | team
----+--------+-------+------
  4 | Robert |    89 | ICE
  3 | Ellen  |    79 | Fire
  2 | Anita  |    45 | UFO
(3 筆資料列)
```

一次寫兩種變數，有升冪和降冪



```
komablog=# select * from users order by team, score desc;
 id | player | score | team
----+--------+-------+------
  1 | 库里   |  28.3 | 勇士
  3 | 阿杜   |  25.6 | 勇士
  2 | 哈登   |  30.2 | 火箭
  6 | 白边   |  19.8 | 热火
  5 | 神龟   |  31.3 | 雷霆
  4 | 阿詹   |  27.8 | 骑士
(6 rows)
```

九、統計抽出數據

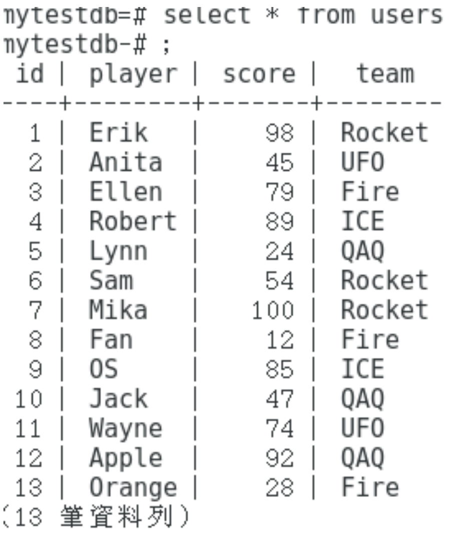
distinct

sum

max/min

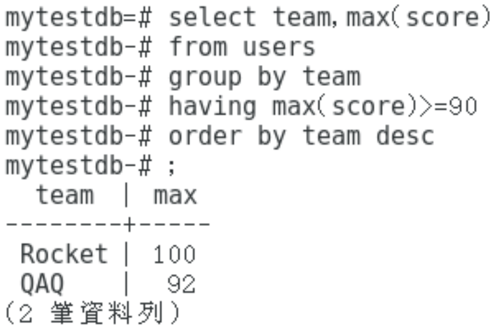
group by/having

增加一些資料～



```
mytestdb=# select * from users
mytestdb-# ;
 id | player | score |  team
----+--------+-------+--------
  1 | Erik   |    98 | Rocket
  2 | Anita  |    45 | UFO
  3 | Ellen  |    79 | Fire
  4 | Robert |    89 | ICE
  5 | Lynn   |    24 | QAQ
  6 | Sam    |    54 | Rocket
  7 | Mika   |   100 | Rocket
  8 | Fan    |    12 | Fire
  9 | OS     |    85 | ICE
 10 | Jack   |    47 | QAQ
 11 | Wayne  |    74 | UFO
 12 | Apple  |    92 | QAQ
 13 | Orange |    28 | Fire
(13 筆資料列)
```

使用 group by + having



```
mytestdb=# select team, max(score)
mytestdb-# from users
mytestdb-# group by team
mytestdb-# having max(score)>=90
mytestdb-# order by team desc
mytestdb-# ;
  team  | max
--------+-----
 Rocket | 100
 QAQ    |  92
(2 筆資料列)
```

十、方便的函數

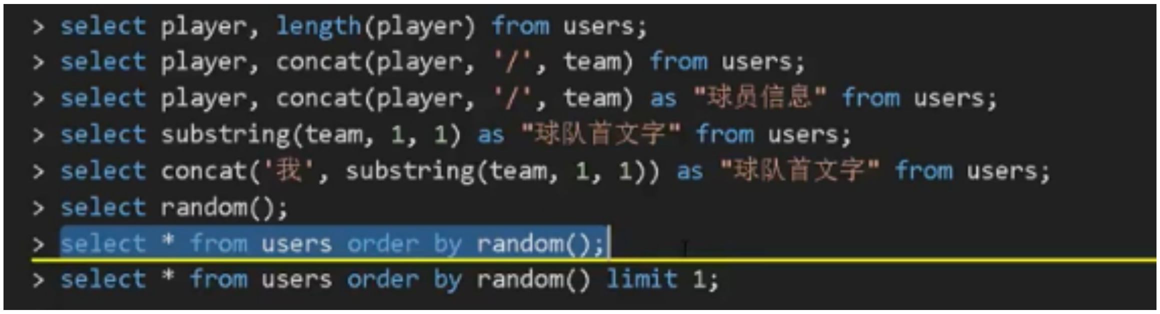
length 長度

concat 組合

alias(as) 別名

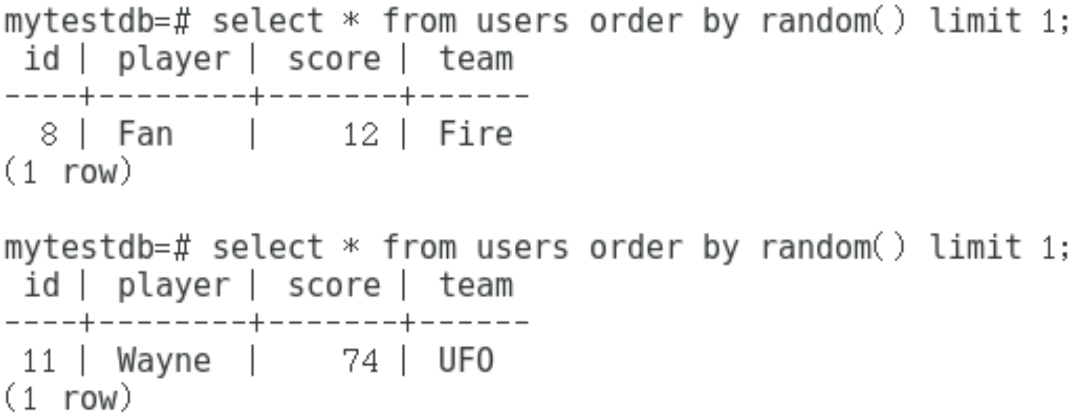
substring 切割

random 隨機數



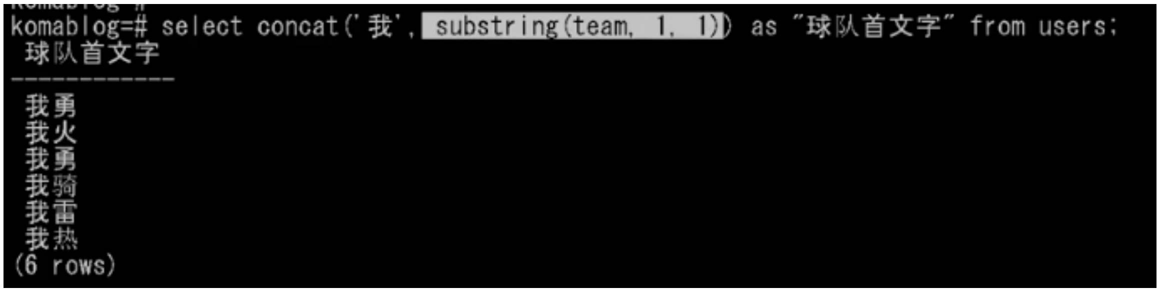
```
> select player, length(player) from users;
> select player, concat(player, '/', team) from users;
> select player, concat(player, '/', team) as "球员信息" from users;
> select substring(team, 1, 1) as "球队首文字" from users;
> select concat('我', substring(team, 1, 1)) as "球队首文字" from users;
> select random();
> select * from users order by random();
> select * from users order by random() limit 1;
```

類似抽獎



```
mytestdb=# select * from users order by random() limit 1;
 id | player | score | team
----+--------+-------+------
  8 | Fan    |    12 | Fire
(1 row)

mytestdb=# select * from users order by random() limit 1;
 id | player | score | team
----+--------+-------+------
 11 | Wayne  |    74 | UFO
(1 row)
```

拼接

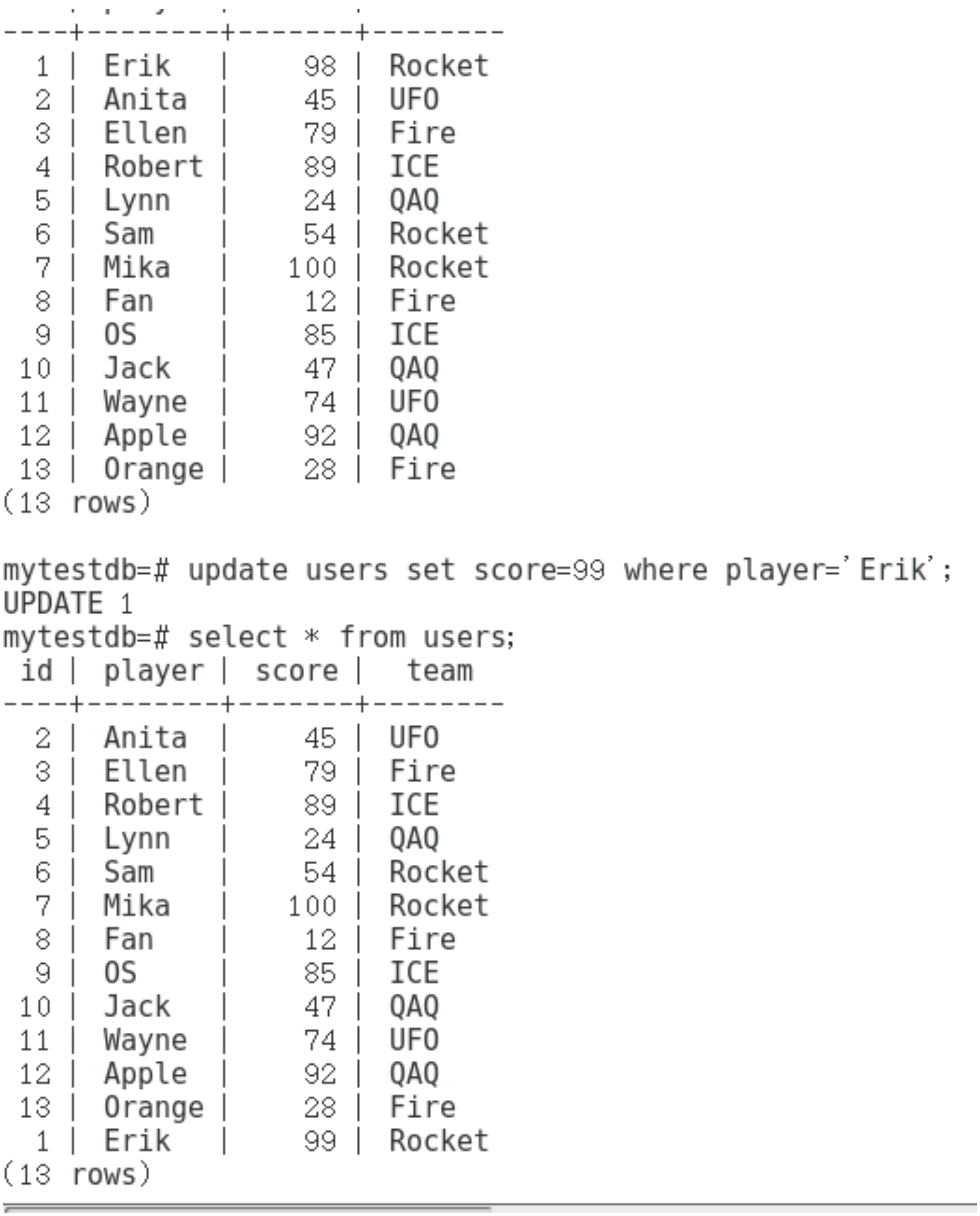


```
komablog=# select concat('我', substring(team, 1, 1)) as "球队首文字" from users;
 球队首文字
------------
 我勇
 我火
 我勇
 我骑
 我雷
 我热
(6 rows)
```

十、更新與刪除

更新數據：

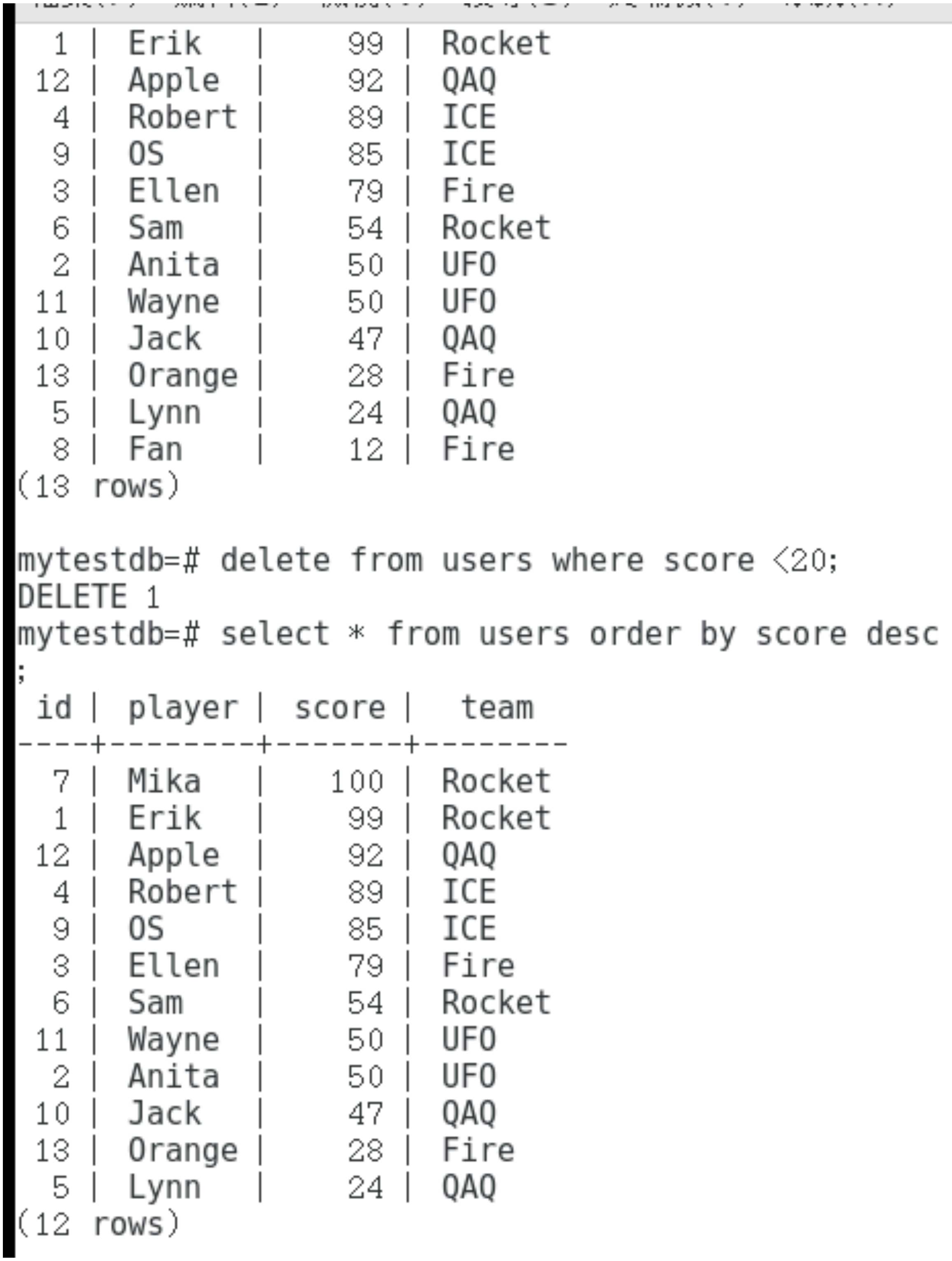
update 表名 set 更改成什麼 where 想要更改的 row



```
----+--------+-------+--------
  1 | Erik   |    98 | Rocket
  2 | Anita  |    45 | UFO
  3 | Ellen  |    79 | Fire
  4 | Robert |    89 | ICE
  5 | Lynn   |    24 | QAQ
  6 | Sam    |    54 | Rocket
  7 | Mika   |   100 | Rocket
  8 | Fan    |    12 | Fire
  9 | OS     |    85 | ICE
 10 | Jack   |    47 | QAQ
 11 | Wayne  |    74 | UFO
 12 | Apple  |    92 | QAQ
 13 | Orange |    28 | Fire
(13 rows)

mytestdb=# update users set score=99 where player='Erik';
UPDATE 1
mytestdb=# select * from users;
 id | player | score |  team
----+--------+-------+--------
  2 | Anita  |    45 | UFO
  3 | Ellen  |    79 | Fire
  4 | Robert |    89 | ICE
  5 | Lynn   |    24 | QAQ
  6 | Sam    |    54 | Rocket
  7 | Mika   |   100 | Rocket
  8 | Fan    |    12 | Fire
  9 | OS     |    85 | ICE
 10 | Jack   |    47 | QAQ
 11 | Wayne  |    74 | UFO
 12 | Apple  |    92 | QAQ
 13 | Orange |    28 | Fire
  1 | Erik   |    99 | Rocket
(13 rows)
```

刪除數據！（危險動作）

delete from 表名 where 條件

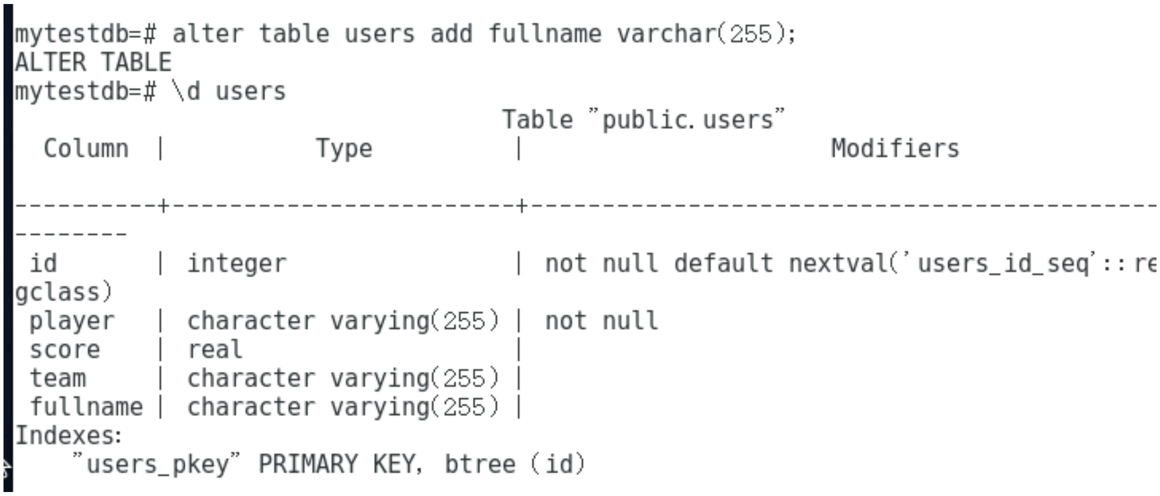


```
  1 | Erik   |    99 | Rocket
 12 | Apple  |    92 | QAQ
  4 | Robert |    89 | ICE
  9 | OS     |    85 | ICE
  3 | Ellen  |    79 | Fire
  6 | Sam    |    54 | Rocket
  2 | Anita  |    50 | UFO
 11 | Wayne  |    50 | UFO
 10 | Jack   |    47 | QAQ
 13 | Orange |    28 | Fire
  5 | Lynn   |    24 | QAQ
  8 | Fan    |    12 | Fire
(13 rows)

mytestdb=# delete from users where score <20;
DELETE 1
mytestdb=# select * from users order by score desc
;
 id | player | score |  team
----+--------+-------+--------
  7 | Mika   |   100 | Rocket
  1 | Erik   |    99 | Rocket
 12 | Apple  |    92 | QAQ
  4 | Robert |    89 | ICE
  9 | OS     |    85 | ICE
  3 | Ellen  |    79 | Fire
  6 | Sam    |    54 | Rocket
 11 | Wayne  |    50 | UFO
  2 | Anita  |    50 | UFO
 10 | Jack   |    47 | QAQ
 13 | Orange |    28 | Fire
  5 | Lynn   |    24 | QAQ
(12 rows)
```

十一、變更表結構

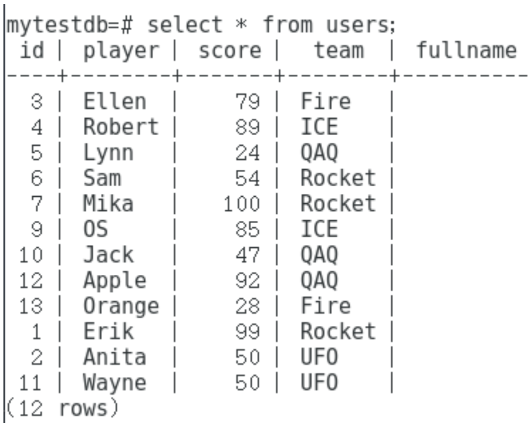
alter 指令：增加一個欄位

alter table 表名 add 欄位名 型態定義



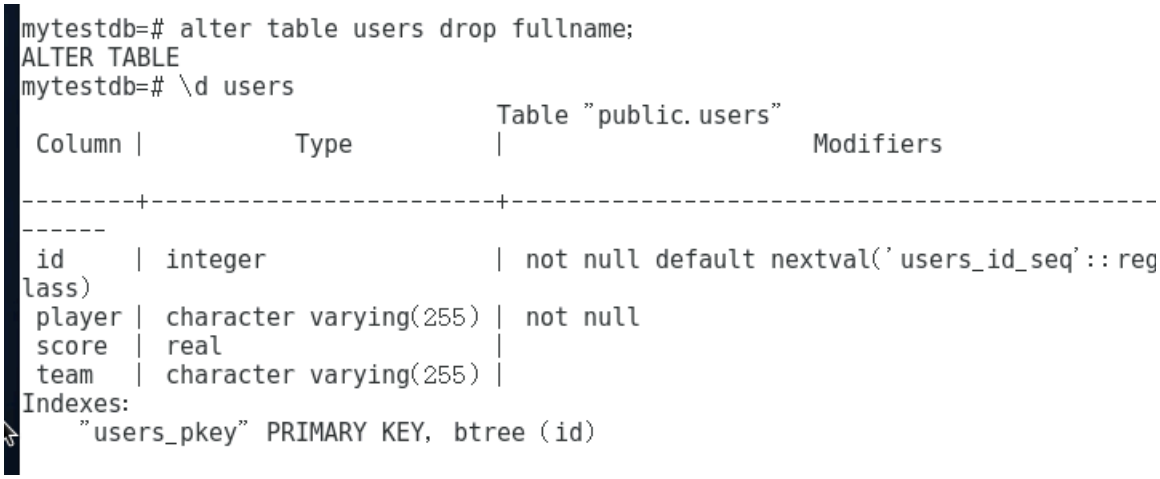
```
mytestdb=# alter table users add fullname varchar(255);
ALTER TABLE
mytestdb=# \d users
                                  Table "public.users"
  Column  |          Type          |                     Modifiers

----------+------------------------+--------------------------------------------
--------
 id       | integer                | not null default nextval('users_id_seq'::re
gclass)
 player   | character varying(255) | not null
 score    | real                   |
 team     | character varying(255) |
 fullname | character varying(255) |
Indexes:
    "users_pkey" PRIMARY KEY, btree (id)
```

看一下表



```
mytestdb=# select * from users;
 id | player | score |  team  | fullname
----+--------+-------+--------+----------
  3 | Ellen  |    79 | Fire   |
  4 | Robert |    89 | ICE    |
  5 | Lynn   |    24 | QAQ    |
  6 | Sam    |    54 | Rocket |
  7 | Mika   |   100 | Rocket |
  9 | OS     |    85 | ICE    |
 10 | Jack   |    47 | QAQ    |
 12 | Apple  |    92 | QAQ    |
 13 | Orange |    28 | Fire   |
  1 | Erik   |    99 | Rocket |
  2 | Anita  |    50 | UFO    |
 11 | Wayne  |    50 | UFO    |
(12 rows)
```

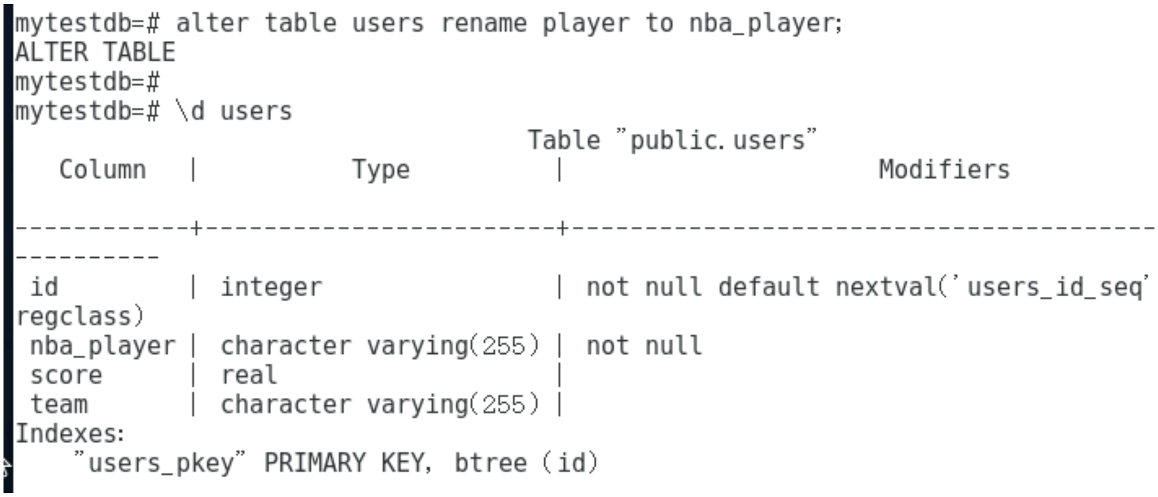
使用 alter 指令 drop 欄位



```
mytestdb=# alter table users drop fullname;
ALTER TABLE
mytestdb=# \d users
                                 Table "public.users"
 Column |          Type          |                     Modifiers

--------+------------------------+---------------------------------------------
------
 id     | integer                | not null default nextval('users_id_seq'::reg
lass)
 player | character varying(255) | not null
 score  | real                   |
 team   | character varying(255) |
Indexes:
    "users_pkey" PRIMARY KEY, btree (id)
```

更改欄位名稱

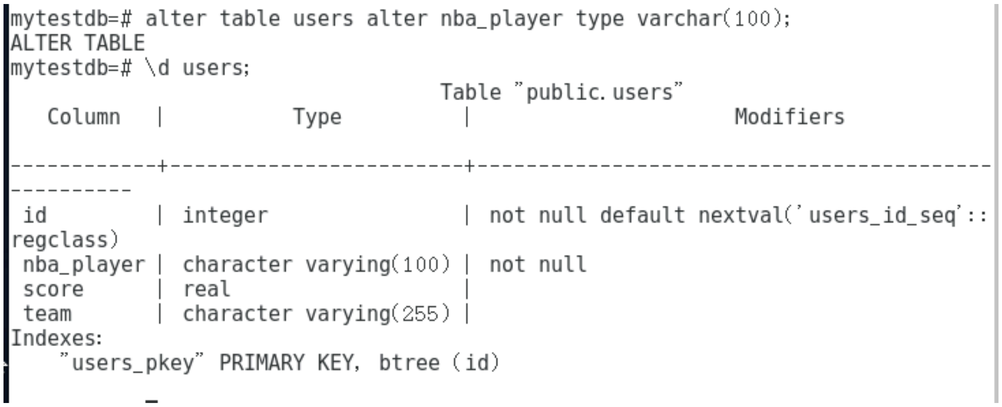
alter table 表名 rename 原欄位名稱 to 新欄位名稱



```
mytestdb=# alter table users rename player to nba_player;
ALTER TABLE
mytestdb=#
mytestdb=# \d users
                                   Table ”public.users”
   Column   |          Type          |                       Modifiers

------------+------------------------+---------------------------------------
----------
 id         | integer                | not null default nextval(’users_id_seq’
regclass)
 nba_player | character varying(255) | not null
 score      | real                   |
 team       | character varying(255) |
Indexes:
    ”users_pkey” PRIMARY KEY, btree (id)
```

更改欄位型態

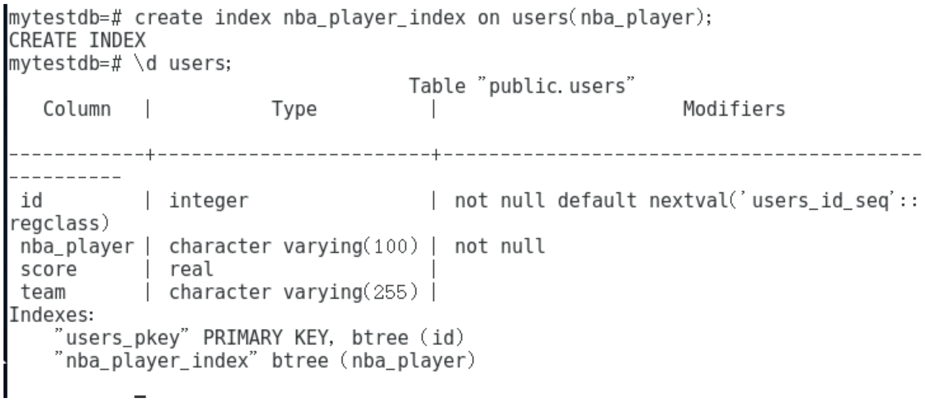
alter table 表名 alter 欄位名稱 type 新欄位型態



```
mytestdb=# alter table users alter nba_player type varchar(100);
ALTER TABLE
mytestdb=# \d users;
                                   Table ”public.users”
   Column   |          Type          |                     Modifiers

------------+------------------------+-----------------------------------------
----------
 id         | integer                | not null default nextval(’users_id_seq’::
regclass)
 nba_player | character varying(100) | not null
 score      | real                   |
 team       | character varying(255) |
Indexes:
    ”users_pkey” PRIMARY KEY, btree (id)
```

加上索引（加速搜尋，但追加數據會影響數據庫的性能，是把雙面刃）

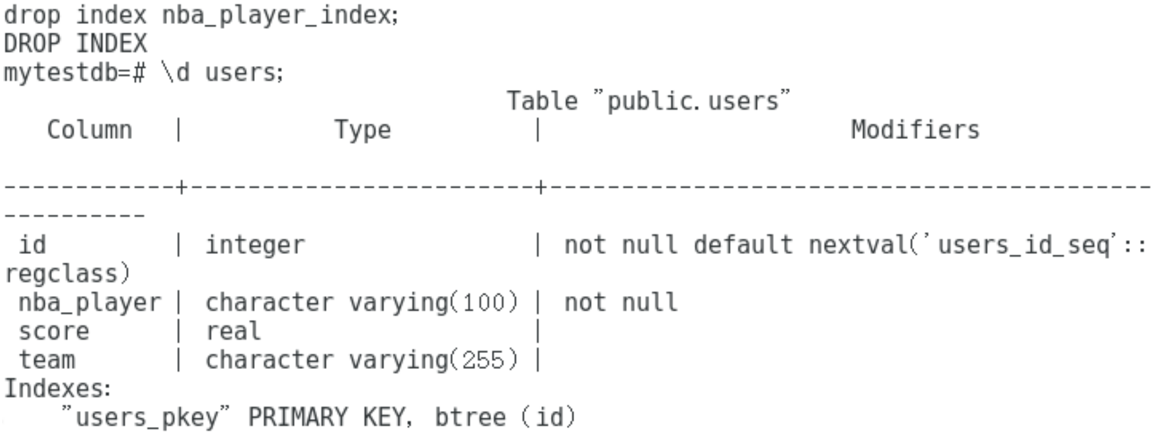
create index 索引名稱 on 表名（欄位名稱）



```
mytestdb=# create index nba_player_index on users(nba_player);
CREATE INDEX
mytestdb=# \d users;
                                   Table ”public.users”
   Column   |          Type          |                     Modifiers

------------+------------------------+-----------------------------------------
----------
 id         | integer                | not null default nextval(’users_id_seq’::
regclass)
 nba_player | character varying(100) | not null
 score      | real                   |
 team       | character varying(255) |
Indexes:
    ”users_pkey” PRIMARY KEY, btree (id)
    ”nba_player_index” btree (nba_player)
```

删除索引

drop index 索引名稱



```
drop index nba_player_index;
DROP INDEX
mytestdb=# \d users;
                                   Table ”public.users”
   Column   |          Type          |                    Modifiers

------------+------------------------+------------------------------------------
----------
 id         | integer                | not null default nextval(’users_id_seq’::
regclass)
 nba_player | character varying(100) | not null
 score      | real                   |
 team       | character varying(255) |
Indexes:
    ”users_pkey” PRIMARY KEY, btree (id)
```