目錄:

Day01(P2~P9)

一、MySQL 概述

二、儲存數據過程

三、MySQL 安裝

Day02(P10~P18)

一、名詞介紹

二、where條件字句(配合查、改、刪操作)

三、表紀錄管理(續)

四、運算符操作(配合查詢、修改、刪除操作)

五、SQL查詢

六、約束

七、普通索引 & 唯一索引

Day03(P18~P29)

一、主鍵索引 & 外鍵索引

二、數據導入

三、數據導出

四、表的複製

五、嵌套查詢

六、多表查詢

七、連接查詢

Day04(P30~P40)

一、數據備份(在linux終端操作)

二、數據恢復(在linux終端操作)

三、ER模型 & ER圖

四、workbench使用(圖形化介面操作數據庫)

五、事務 & 事務回滾

六、Python數據庫編程

MySQL進階I:

Day 01:

一、MySQL 概述

1. 甚麼是數據庫?

數據庫是一個儲存數據的倉庫

2. 哪些公司在用數據庫

金融機構、購物網站、遊戲網站、論壇網站......

3. 提供數據庫服務的軟件

(1) 軟件分類

MySQL、SQL\_Server、Oracle、DB2、Mariadb、MongoDB...

(2) 在生產環境下，如何選擇使用哪個數據庫軟件

(a) 是否開源

˙ 開源軟件

(開源的意思:原代碼開放代表此軟件可修改、複製、傳播，但不一 定免費)

Ex:

MySQL、Mariadb、MongoDB

˙ 商業軟件

Ex:

Oracle、DB2、SQL\_Server

(b) 是否跨平台

˙ SQL\_Server 不跨平台，其他的都屬於跨平台的

˙ 公司類型

商業軟件: 政府部門、金融機構

開源軟件: 遊戲網站、購物網站、論壇網站...

4. MySQL特點

(1) 關係型數據庫

特點 :

1. 數據是以行和列的形式儲存的

2. 這一系列的行和列稱為表

3. 表中的每一行叫一條紀錄

4. 表中的每一列叫一個字段

5. 表和表之間的邏輯關聯叫關係

6. 關係型數據庫的核心內容是: 關係 即 二維表

Ex:(關係型數據庫儲存)

表1、學生訊息表

姓名 年齡 班級

Eric 18 AID1712

Iris 22 AID1711

表2、班級訊息表

班級 班主任

AID1712 侯大大

AID1711 孫大大

Ex:(非關係型數據庫儲存)

{姓名:'Eric',年齡:18,班級:'AID1712',班主任:'侯大大'}

(2) 跨平台

可以在Unix、Linux、Windows上運行MySQL服務

(3) 支持多種編成語言

Python、Java、PHP、......

二、儲存數據過程

舉例一個用Python語言寫的天貓購物網站，存取流程如下:

加入購物車 -> buy.py -> MySQL數據庫服務器

查看購物車 -> show.py -> MySQL數據庫服務器

...

最後以web頁面形式回傳

三、MySQL 安裝

1. Ubuntu安裝MySQL服務

˙ 安裝服務端

指令: sudo apt-get install mysql-server

˙ 安裝客戶端

指令: sudo apt-get install mysql-client

2. Window安裝MySQL服務

˙ 下載MySQL安裝包(Windows)

mysql-install-\*\*5.7\*\*.msi

˙ 雙擊、按照教學安裝即可

3. 啟動和連接MySQL服務

1. 服務端啟動

˙ 查看MySQL服務的狀態

sudo /etc/init.d/mysql status

˙ 啟動MySQL服務

sudo /etc/init.d/mysql start

˙ 停止MySQL服務

sudo /etc/init.d/mysql stop

˙ 重啟MySQL服務

sudo /etc/init.d/mysql restart

2. 客戶端連接

1. 命令格式

˙ mysql -h主機名 -u用戶名 -p密碼

mysql -hlocalhost -uroot -pa123456

2. 本地連接可以省略 -h 選項 否則要加上ip位置

3. 斷開與mysql連接

exit | quit | \q

4. 基本的SQL命令 (詳見在外)

1. 使用規則

˙ 每條命令必須以分號(;)結尾

show databases;

˙ SQL命令不區分字母大小寫

ShoW DaTABaseS;

˙ 使用 \c 終止命令執行

show databases\c

(注意若\c打在分號後面沒有意義)

2. 庫的管理

1. 庫的基本操作:

˙查看已有的庫

show databases;

˙創建庫[指定字符集,默認為latin1]

create database 庫名 [default charset=utf8];

˙查看創建庫的語句

show create database 庫名;

˙查看當前所在庫

select database();

˙切換庫

use 庫名;

˙查看庫中已有的表

show tables;

˙刪除庫

drop database 庫名;

3. 庫的命名規則:

˙可以使用數字、字母、下滑線，但是不能是純數字

˙庫名具有唯一性

˙不能使用特殊字符和mysql關鍵字

＊練習

˙創建庫 AID1712db,指定字符集為utf8

˙進入到庫 AID1712db中

˙查看當前所在庫

˙查看庫中已有的表

˙查看AID1712db的字符集

˙刪除庫AID1712db

3. 表的管理

1. 表的基本操作

˙創建表

create table 表名(字段名 數據類型,字段名 數據類型, ...);

create table stu(id int,name char(10),age int);

(上面的id name age屬於字段)

˙查看創建表的語句

show create table 表名;

˙查看表結構

desc 表名;

˙刪除表

drop table 表名;

＊ 注意:

所有的數據都是以文件的形式儲存在數據庫目錄下

(數據庫目錄: /var/lib/mysql 需要以root權限)

＊ 練習

˙創建庫python

˙在python庫中創建表py\_mysql,字段有如下三個

id kuname biaoname 數據類型自己定義

˙查看創建表的語句

˙查看py\_mysql的表結構

˙刪除表py\_mysql

4. 表紀錄的管理

˙在表中插入紀錄

insert into 表名 values(值1),(值2),...;

(上面的值1,值2...是屬於紀錄)

˙查看表的紀錄

select \* from 表名;

select 字段名1,字段名2,...from 表名;

＊ 練習

˙查看所有的庫

˙創建一個新庫studb

˙在studb中創建一張表t1,字段有id name age score

數據類型自定義

˙查看t1的表結構

˙在表中隨便插入兩條紀錄

˙查看t1表中的所有紀錄

˙查看創建表t1的語句(字符集)

5. 如何更改默認字符集

1. 方法

˙ 通過更改Mysql的配置文件

2. 步驟

˙先獲取root權限

sudo -i

˙修改配置文件

vi /etc/mysql/mysql.conf.d/mysqld.cnf

在[mysqld]加上

character\_set\_server = utf8

3. 重啟mysql服務

sudo /etc/init.d/mysql restart

6. 客戶端把數據儲存到數據庫服務器上的過程

1. 連接到數據庫服務器: mysql -uroot -p密碼

2. 選擇庫: use 庫名;

3. 創建/修改表

4. 斷開與數據庫的連接: exit | quit | \q

7. 數據類型

1. 數值類型(有符號signed 和 無符號unsigned)

(1) 整型

˙int 大整型(4Bytes)

取值範圍: 0 ~ 2\*\*32-1

˙tinyint 微小整型(1Byte)

取值範圍: 有符號的(signed) -128 ~ 127

無符號的(unsigned) 0 ~ 255

˙smallint 小整型(2Bytes)

取值範圍: 0 ~ 65535

˙bigint 極大整型(8Bytes)

取值範圍: 0 ~ 2\*\*64-1

(2) 浮點型

˙float(4Bytes,最多顯示7個有效位)

字段名 float(m,n) m->總位數 n->小數位位數

(＊ m + n <= 7，超過7位後的數值會亂顯示)

Ex: float(5,2) 取值範圍: -999.99 ~ 999.99

＊ 浮點型插入整數時，會自動補齊小數位數

小數位如果多於指定位數，會對下一位四捨五入

˙double(8Bytes,最多顯示15個有效位)

字段名 double(m,n)

˙decimal(M+2Bytes,最多可以顯示28個有效位)

decimal(M,D)

2. 字符類型

˙char(寬度範圍 ＊默認為1)

寬度取值範圍: 0 ~ 255

name char(5)

˙varchar(變動長度)

取值範圍: 1 ~ 65535

＊varchar 沒有默認寬度，必須給定一個寬度

char和varchar使用時都給定寬度，但不能超過各自範圍

˙char 和 varchar 的特點

1. char 浪費儲存空間，但性能高

2. varchar 節省儲存空間，但性能低

˙字符類型的寬度和數值類型的寬度區別

1. 數值類型的寬度為顯示寬度，只用於select查詢時使用

和占用儲存空間大小無關，可用zerofill查看效果

2. 字符類型的寬度超過則無法儲存

＊練習

˙創建表stuinfo1712,utf8,字段要求:

學號: id 要求顯示寬度為3，位置不夠用0填空

姓名: name 變長，寬度20

年齡: age 微小整型，不能輸入負數

身高: height 浮點型，小數位數2位

工資: salary 浮點型，小數位2位 最大值99999.99

˙在表中插入兩條紀錄

˙查詢表中記錄，只顯示姓名，年齡和工資

˙查看表結構

3. 枚舉類型

˙定義:

字段值只能在列舉的範圍內選擇

˙enum 選項 (單選,最多只能有65535個選項)

字段名 enum(值1,值2,...)

˙set 選項 (多選,最多只能有64個選項)

字段名 set(值1,值2,...)

likes set("Study",'Girl','Python','MySQL')

選項裡必須要直接寫"值1 值2..."

4. 日期時間類型

˙year: 年 YYYY

˙date: 日期 YYYYMMDD

˙time: 時間 HHMMSS

˙datetime: 日期時間 YYYYMMDDHHMMSS

˙timestamp: 日期時間 YYYYMMDDHHMMSS

(datetime和timestamp差別於默認值前者為NULL後者為現在時間)

8. 表字段的操作

1.語法: alter table 表名 執行動作;

˙添加字段

alter table 表名 add 字段名 數據類型 [放入位置];

加在第一個位置: first

加到指定位置: after 前一個字段名

˙刪除字段(drop)

alter table 表名 drop 字段名;

˙修改數據類型(modify)

alter table 表名 modify 字段名 數據類型;

˙修改字段名

alter table 表名 change 舊名 新名 數據類型;

˙修改表名

alter table 表名 rename 新表名;

＊作業:

填空題:

˙MySQL中的數據類型有\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_

˙關係型數據庫的核心內容是 \_\_ 即 \_\_

簡答題:

˙簡述客戶端把數據儲存到數據庫服務器上的過程

˙char varchar區別 各自的特點

操作題:

˙創建一個庫school

˙在庫中創建表student來儲存學生訊息

˙字段id 要求顯示寬度為3位，不夠用0填充

姓名name、年齡age只能為正數、成績score浮點

性別sex單選、愛好likes多選、入學時間(年月日)

˙查看students的表結構

˙在表中增加一個字段id，加在第一列

˙表中任意插入5條紀錄

˙查看所有學生的姓名、成績、入學時間

Day01 End.

MySQL進階II:

Day 02:

一、名詞介紹

1. DB(Database)

DB舊式數據庫，儲存數據的倉庫

2. DBMS(Database Management System)

數據庫管理系統

管理數據庫的軟件，MySQL、Oracle...

3. DBS(Database System)

數據庫系統

DBS = DB(儲存)+DBMS(數據庫軟件)+數據庫應用(財務管理系統，人事 管理系統)+用戶

二、where條件字句(配合查、改、刪操作)

1. 語法格式

select \* from 表名 where 條件;

三、表紀錄管理(續)

1. 更新表紀錄

update 表名 set 字段名=值,字段名=值2,...where 條件;

＊如果不加上where條件會導致表內所有的紀錄都改變了

2. 刪除表紀錄

delete from 表名 where 條件;

＊如果不加上where條件會導致表內所有的紀錄都刪除

＊練習 <- 在SANGUO 庫裡 和 MoShou 庫裡

˙查找所有蜀國人訊息

˙查找女英雄訊息，只顯示姓名和性別和國家

˙把魏延性別改為女，國籍改泰國

˙把id為2的紀錄名字改為司馬懿，性別男、國家魏國

˙刪除所有泰國人

四、運算符操作(配合查詢、修改、刪除操作)

1. 數值比較 & 字符比較

1. 數值比較運算符: = != > >= < <=

2. 字符比較運算符: = !=

3. 語法格式

查詢:

˙select \* from 表名 where 字段名 運算符 數字/字符;

修改:

˙update 表名 set 字段名=值,...

where 字段名 運算符 數字/字符;

刪除:

˙delete from 表名 where 字段名 運算符 數字/字符;

＊練習

˙找出攻擊值高於150的英雄名字和攻擊值

˙將趙雲的攻擊值改成666，防禦88

2. 邏輯比較:

1. 運算符:

and(多個條件同時滿足)

or(多個條件一個滿足)

＊練習

˙找出攻擊值大於200的蜀國英雄名字和攻擊值

˙將吳國英雄中攻擊值為110的英雄改為100,防禦設為60

˙查找魏國和吳國的英雄訊息

3. 範圍內比較

1. 運算符: between and / in / not in

2. 語法格式:

查詢語句 字段名 between 值1 and 值2

查詢語句 字段名 in (值1,值2,...)

查詢語句 字段名 not in (值1,值2,...)

＊練習

˙查找攻擊值100~200之間的蜀國英雄訊息

˙查找id在1,3,5,7中的英雄id姓名

˙查找蜀國和吳國以外的國家女英雄

˙查找編號1或3或5的蜀國英雄和貂蟬的編號、姓名、國家

4. 匹配空、非空

1. 空: is null

2. 非空: is not null

＊練習

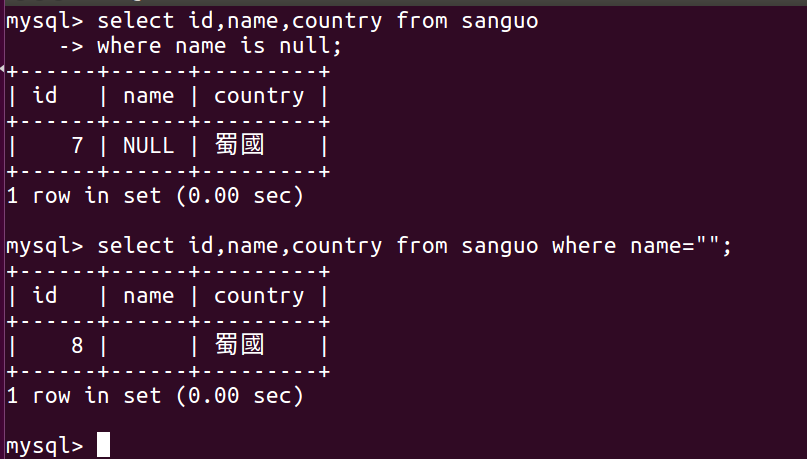
˙查找姓名為NULL蜀國女英雄訊息

˙查找姓名為""的英雄id、姓名、國家

＊注意

null 是空值，必須用 is 或 is not 去匹配

"" 是空字符串，用 = 或者 != 去匹配



3. 模糊比較

1. 語法格式

˙字段名 like 表達式

2. 表達式

˙ \_ : 匹配單個字符

˙ % : 匹配0到多個字符

＊練習

select id,name from sanguo where name like "\_%\_";

## 名字中至少有兩個字符

select id,name from sanguo where name like "%";

## 名字不為null的所有紀錄

select id,name from sanguo where name like "\_\_\_";

## 名字中只有三個字符

select id,name from sanguo where name like "趙%";

## 名字中第一個只能是趙的字符

(相似於在尋找檔案的方式)

五、SQL查詢

1. 總結(執行順序)

3. select ... 聚合函數 from ...

1. where ...

2. group by ...

4. having ...

5. order by ...

6. limit ... ;

˙ order by

作用: 對查詢的結果進行排序

語法: 查詢語句 order by 字段名 排序方式;

排序方式:

1. ASC(默認) 升序

2. DESC 降序

＊練習

˙將英雄按防禦值從低到高排序

˙將蜀國英雄按攻擊值從高到低排序

˙將魏蜀兩國男英雄中名字為三個字的英雄按防禦值升序

˙limit (永遠放在SQL語句的最後面)

作用: 限制顯示查詢紀錄的條數

用法:

limit n --> 顯示 n 條紀錄

limit m,n --> 從第 m+1 條紀錄開始顯示 n 條

(default m == 0)

linit 2,4 --> 從第 3 條紀錄開始顯示 3,4,5,6 條紀錄

＊練習

˙查找防禦值倒數第二名至倒數第四名的蜀國英雄紀錄

˙查找攻擊值前三名且名字不為空值的蜀國英雄的姓名

顯示攻擊值和國家

˙聚合函數

1. 分類:

1. avg(字段名) [as 別名]: 求字段裡記錄的平均值

2. sum(字段名) [as 別名]: 求字段裡記錄的和

3. max(字段名) [as 別名]: 求字段裡記錄的最大值

4. min(字段名) [as 別名]: 求字段裡記錄的最小值

5. count(字段名) [as 別名]: 統計字段紀錄的個數

例: select max(gongji) as matt from sanguo;

＊練習

˙攻擊力最強值是多少

˙統計表中id,name字段分別有多少條紀錄

＊注意: 空值NULL不會被count統計，但空字符串""可以被統計

˙計算蜀國英雄的總攻擊力

˙統計蜀國英雄中攻擊值大於200的英雄數量

＊使用count(\*)可以統計到NULL空值，否則空值會被忽略

˙group by

作用: 給查詢的結果進行分組

(先分組再聚合)

＊練習:

˙查詢sanguo表中一共有幾個國家

select country from sanguo group by country;

˙計算所有國家的平均攻擊力

select country,avg(gongji) from sanguo

group by country;

˙查詢所有國家中英雄數量最多的前兩名的國家名稱和英雄數量

＊注意

˙group by 之後的字段必須要為select之後的字段

select 字段名1 from 表名 group by 字段名1;

˙如果select之後的字段不一致，則必須要對select之後的

字段做聚合處理

˙having

作用: 對查詢的結果做進一步地塞選

＊練習

˙找出平均攻擊力大於105的國家前兩名，顯示國家名和平均攻 擊力

select avg(gongji),country from sanguo

group by country

having avg(gongji)>105

order by avg(gongji) desc

limit 2;

＊注意

˙having語句通常與group by語句聯合使用，用來過濾由group by語句傳回紀錄集

˙having語句彌補了where關鍵字不能與聚合函數使用的不足， having操作的是聚合函數生成的顯示列

˙distinct

作用: 不顯示字段的重複值

＊練習

˙sanguo表中一共有多少個國家

select distinct 字段名1[,字段名2...] from 表名;

˙蜀國一共有多少個英雄

＊注意

˙distinct處理的是distinct和from之間的所有字段，所有字 段值必須全部相同才能去重

˙distinct不能對任何字段做聚合處理

˙查詢表紀錄時做數學運算

運算符: + - \* / %

＊練習:

˙查詢時顯示所有英雄攻擊力\*10

(運算符直接加在字段值後面就可以了)

˙查詢時顯示所有英雄的防禦力+5

六、約束

作用: 為了保證數據表中數據的完整性、一致性、有效性

分類:

1. 默認約束(default)

˙插入紀錄時，如果不給該字段賦值，則使用默認值

˙格式: 字段名 數據類型 default 值;

2. 非空約束(not null)

˙作用: 不允許將該字段設置為NULL

˙格式: 字段名 數據類型 not null [default 值];

七、普通索引 & 唯一索引

定義: 對數據中表的一列或多列的值進行排序的一種結構

(MySQL中用Btree方式)

＊Btree: 則是分組逐次判斷條件的樹狀數據結構

(MySQL中只支持分三層)

優點: 加快數據的查詢速度

缺點:

1. 當對表中的數據進行增加、刪除、修改的時候，索引也到動態維 護，降低了數據的維護速度

(新增了一個數據，代表其餘數據索引都會變動)

2. 索引需要占用物理空間

索引分類:

1. 普通索引(index):

˙使用規則

1. 一個表中可以有多個index字段

2. 字段的值可以有重複，且可以為NULL值

3. 經常把做查詢條件的字段設置為index字段

4. index字段的key標誌是MUL

˙創建index

1. 創建表時創建

˙ index(字段名1),index(字段名2),...

2. 在已有表中創建index

˙ create index 索引名 on 表名(字段名);

＊注意: 索引名一般和字段名一樣

3. 查看普通索引

˙desc 表名; --> 查看key標誌

˙show index from 表名;

4. 刪除索引

˙ drop index 索引名 on 表名;

2. 唯一索引(unique key):

˙使用規則

1. 一個表中可以有多個unique字段

2. unique字段的值不允許重複，但可以為NULL

3. unique的key標誌是UNI

˙創建唯一索引unique

1. 創建表時創建

˙unique(字段名1),unique(字段名2),...

˙字段名 數據類型 unique

2. 在已有表中創建

˙create unique index 索引名 on 表名;

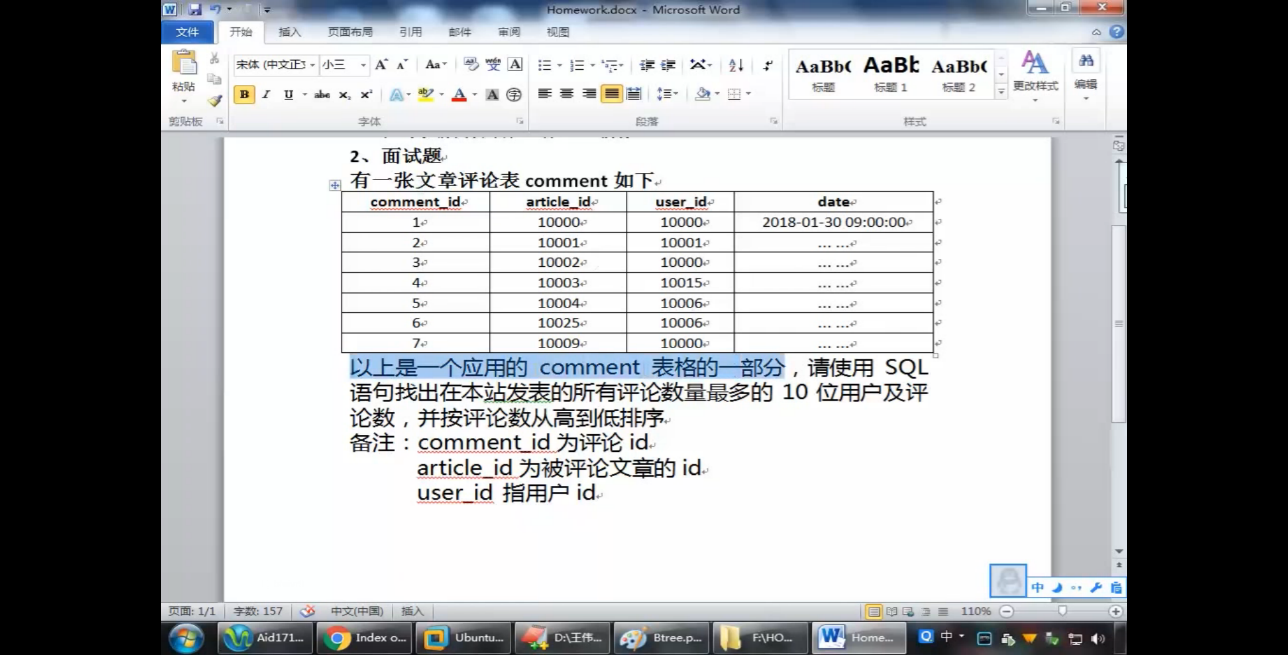
3. 刪除唯一索引

˙drop index 索引名 on 表名;

＊注意: 在刪除時只能一個一個刪

面試題:

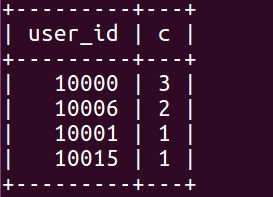
在MySQL Day02面試題



解答:

select user\_id,count(user\_id) as c from interview

group by user\_id order by c desc limit 10;



Day02 End.

MySQL進階III:

Day 03:

一、主鍵索引 & 外鍵索引

1. 普通索引(index)

2. 唯一索引(unique index)

3. 主鍵索引(primary index)

˙使用規則

1. 一個表中只能有一個主鍵(primary)字段

2. 對應字段的值不允許重複且不可為空

3. 主鍵字段的key標誌為PRI

4. 把表中能夠唯一標示一條紀錄的字段設置為主鍵，

通常把表中紀錄編號的字段設置為主鍵

˙創建主鍵(primary key)

1. 創建表時創建主鍵

˙字段名 數據類型 primary key;

˙primary key(字段名[,name]]) # 要加在最後面

[也可以用複合主鍵]

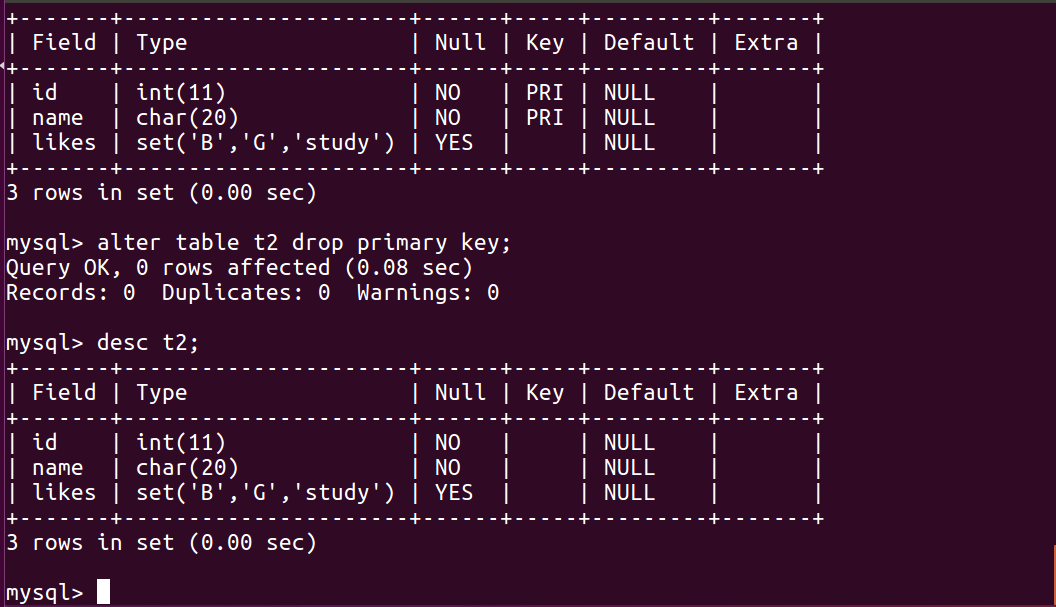
2. 在已有表中創建

˙alter table 表名 add primary key(字段名);

3. 刪除主鍵

˙alter table 表名 drop primary key;

＊即使主鍵被刪除了但是數據依然不可以為空值



(見:MySQL03 primary key圖檔)

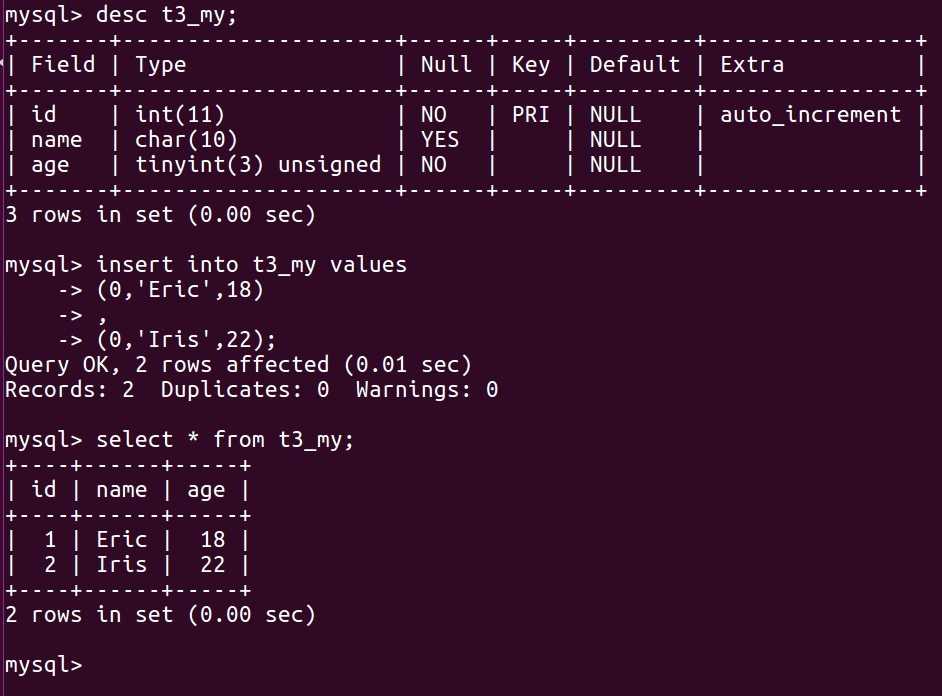
4. 自增長屬性(auto\_increment)

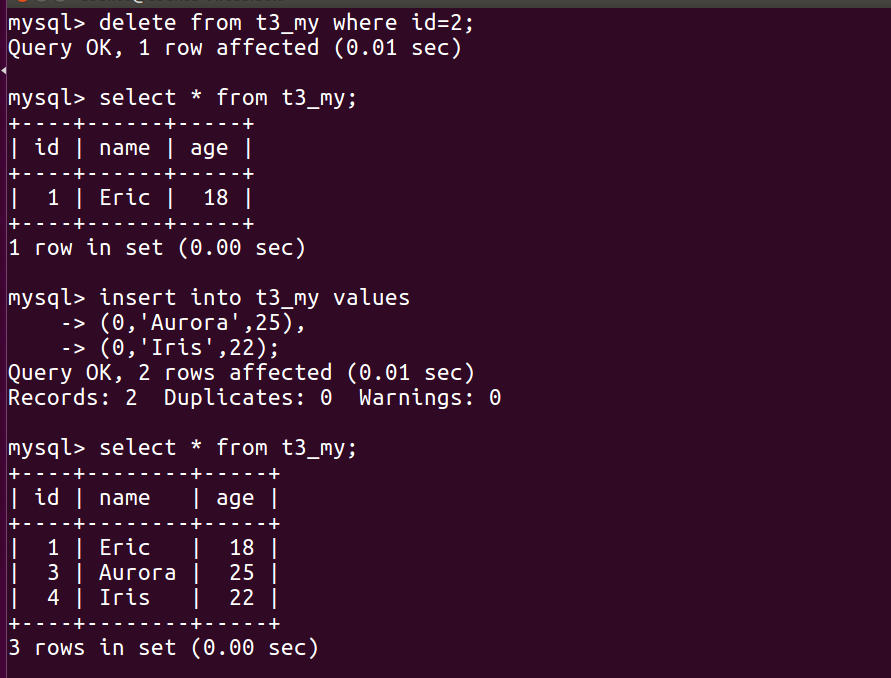
作用: 通常和主鍵字段一起配合使用

1. 創建表時創建

˙字段名 數據類型 primary key auto\_increment

＊數據會依序排列，即使數據被刪除過後仍然會依刪除 前的數據排列





(見:MySQL03 auto\_increment[全])

2. 在已有表中增加自增長屬性

˙alter table 表名 modify 字段名 數據類型

primary key auto\_increment

3. 刪除自增長屬性

˙alter table 表名 modify 字段名 數據類型;

＊要先刪除自增長屬性才有辦法刪除主鍵

4. 外鍵索引(foreign key)

1. 定義: 讓當前表的字段值在另一個表的範圍內選擇

2. 語法格式:

˙foreign key(參考字段名)

references 被參考表名(被參考字段名) #被參考為主，參考為從

on delete 級聯動作

on update 級聯動作

3. 舉例:

# 已經繳過錢的學生

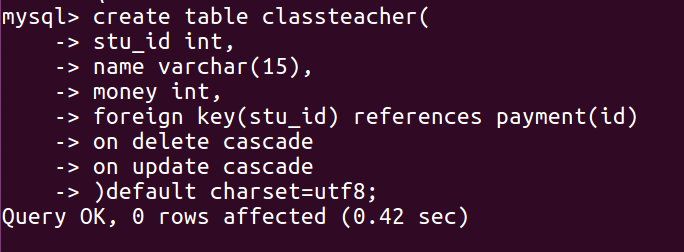
表1: 繳費訊息表(財務) (db3的payment表)

學號 姓名 班級 繳費金額

1 Eric AID1712 25000

2 Iris AID1712 20000

表2: 學生訊息表(班主任) (db3的classteacher表 #圖[創建方法])



學號 姓名 繳費金額

1 Eric 25000

3 Aurora 28000 # <-無法插入此紀錄

＊若在表1的內容裡沒有表2的學生，表2要插入紀錄則無法插入

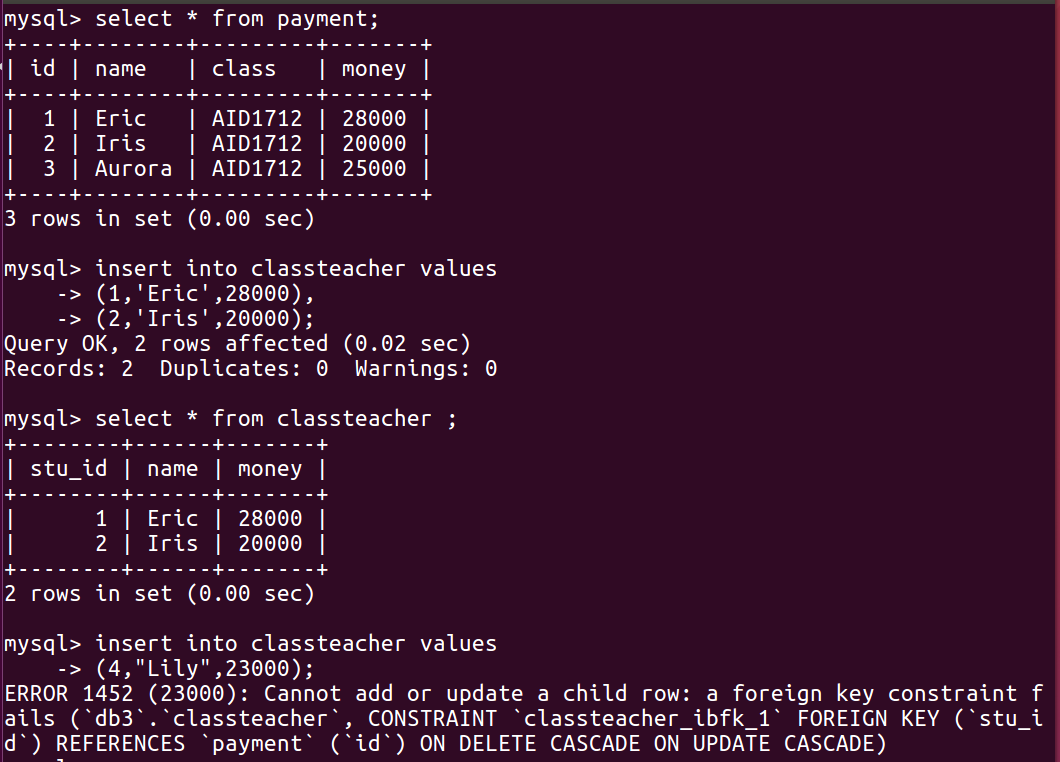
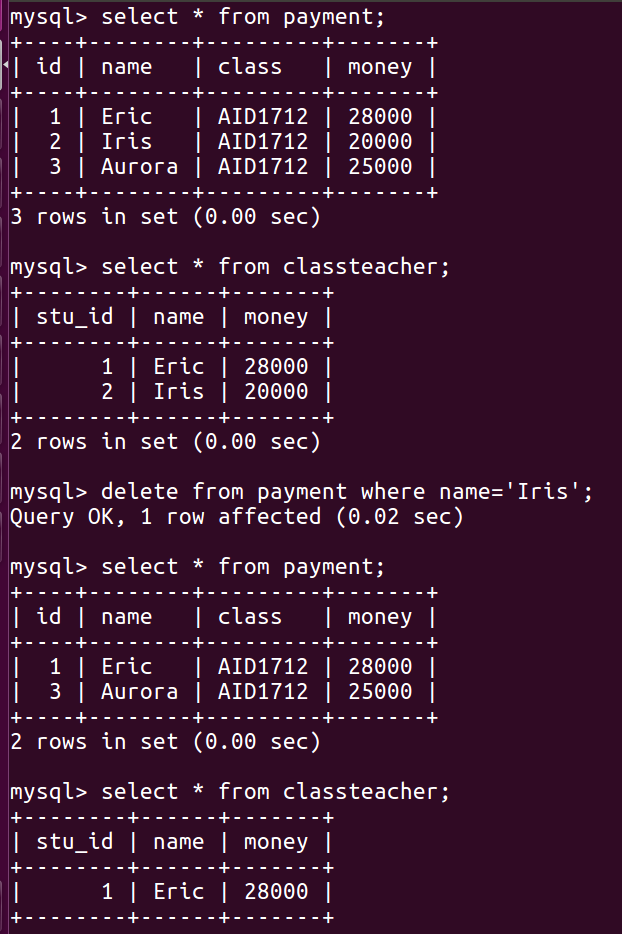


表2無法插入紀錄

表1

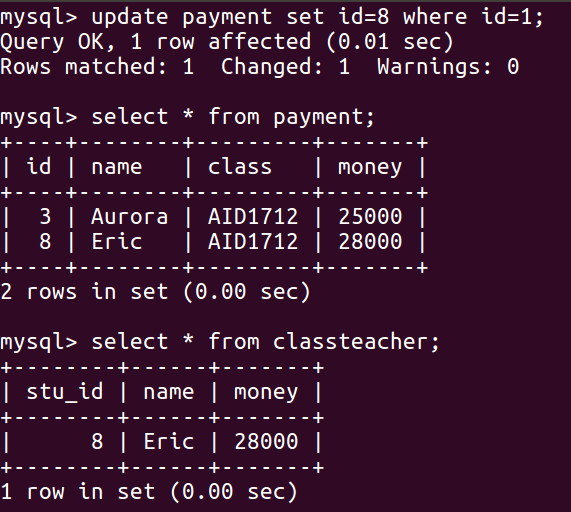
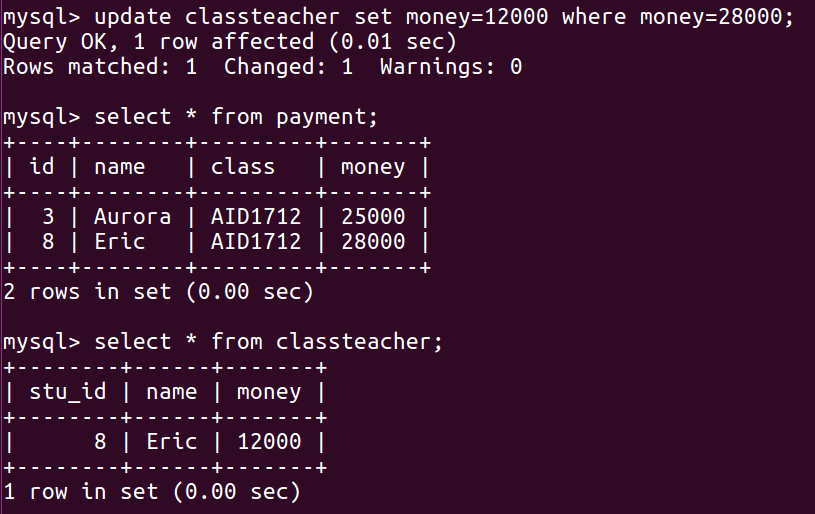
(見:MySQL03 foreign result圖)

＊若刪除表1數據，則表2的數據也會被刪除



(見:MySQL03 foreign result02圖)

＊只更改"被參考的字段"(references的字段)才可以全部被更新

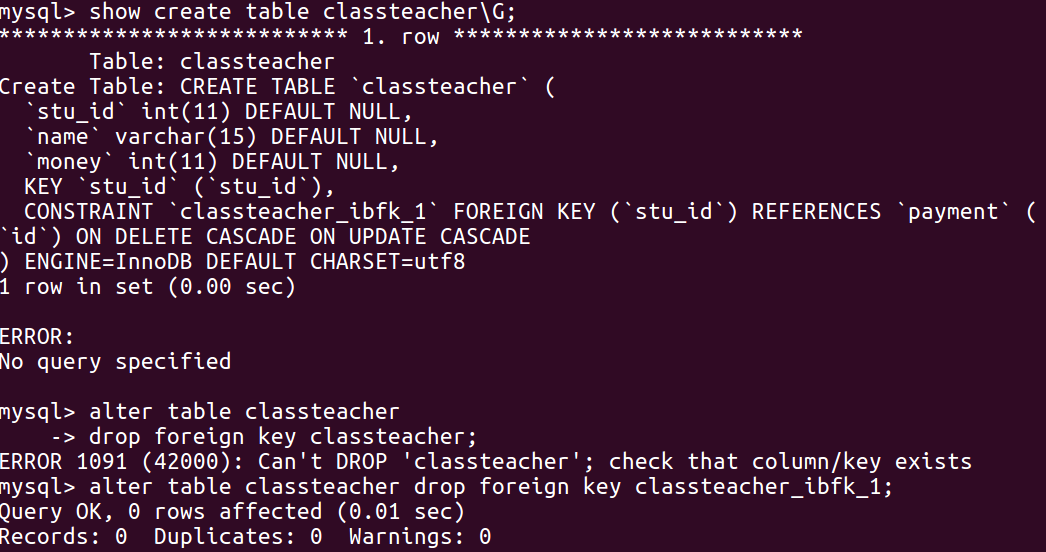
(見:MySQL03 foreign update[02]比對圖)

4. 刪除外鍵

˙alter table 從表名 drop foreign key 外鍵名;

外鍵名的查看方式

˙show create table 表名;



(見:MySQL03 delete foreign method圖)

＊若要刪除表2數據必須把外鍵給刪除，否則會受到外鍵的條件限制

5. 在已有表中添加外鍵

1.語法格式:

˙alter table 表名 add

foreign key(參考字段名)

references (被參考字段名)

on delete 級聯動作

on update 級聯動作

＊在已有表中添加外鍵時，會受到表中原有數據的限制

6. 級聯動作

˙cascade: 數據級聯更新

主表 = 被參考表(references 後面的表和字段)

從表 = 參考表(foreign key 後面的字段)

1. 當主表刪除紀錄時，若從表有相關紀錄則級聯刪除

2. 當主表更新被參考字段的值時，從表則級聯更新參考 字段的值

˙restrict(默認)

1. 當主表刪除/更新紀錄時，若從表中有相關紀錄則不允許 刪除/更新

˙set null

1. 當主表刪除/更新紀錄時，若從表中相關紀錄外鍵字段值 變為null

˙no action

同restrict，都是立即檢查外鍵限制

7. 使用規則

1. 兩張表被參考字段和參考字段的數據類型要一致

2. 被參考字段必須是KEY的一種，通常是primary key

二、數據導入

1. 作用: 將文件系統的內容導入到數據庫中

2. 語法格式:

˙load data infile "文件名(要打絕對路徑)"

into table 表名

fields terminated by "分隔符"

lines terminated by "分隔符"

舉例:

ubuntu: x :1000:1000:Ubuntu,,,:/home/ubuntu:/bin/bash

用戶名:密碼: UID: GID:描述 :家目錄 :登入權限

3. 把/etc/passwd 導入到mysql數據庫中

4. 操作步驟(在db3裡的userinfo)

1. 在數據庫中創建對應的表

2. 查看數據庫的默認搜索路徑

˙show variables like "secure\_file\_priv";

(若忘記了搜索內容可以用模糊查詢)

3. 將系統文件拷貝到數據庫的默認搜索路徑中

˙sudo cp /etc/passwd /var/lib/mysql-files/

4.導入文件

˙load data infile "/var/lib/mysql-files/passwd"

into table userinfo

fields terminated by ":"

lines terminated by "\n";

三、數據導出

1. 作用: 將數據庫中表的紀錄保存到系統文件裡面

2. 語法格式

˙select ... from 表名(也可以加上當前庫名 Ex: mysql.user)

into outfile "文件名(要打絕對路徑)"

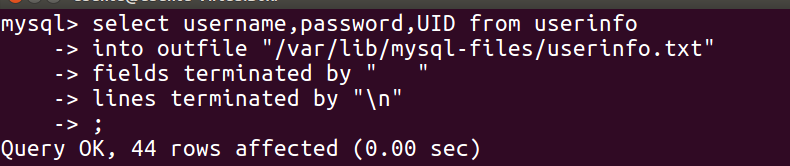
fields terminated by "分隔符"

lines terminated by "分隔符";

＊練習

˙把userinfo表中的用戶名、密碼、UID三個字段導入到userinfo.txt

(查詢userinfo.txt文件) # 文件不須事先創建



(見:MySQL03 Outfile data method圖)

˙將庫名:mysql中user表中的User,Host兩個字段的值導出到user2.txt

＊注意

1. 導出的內容由SQL查詢語句決定

2. 執行導出命令時路徑必須指定在對應的數據庫目錄下

四、表的複製

1. 語法格式:

˙create table 表名 select 查詢命令;

(以userinfo為例 複製到 userinfo2)

＊練習:

˙複製userinfo表的前10行，命名為userinfo3

(~ 後面加上limit 10)

˙複製userinfo表的用戶名、密碼、UID的2~10條紀錄

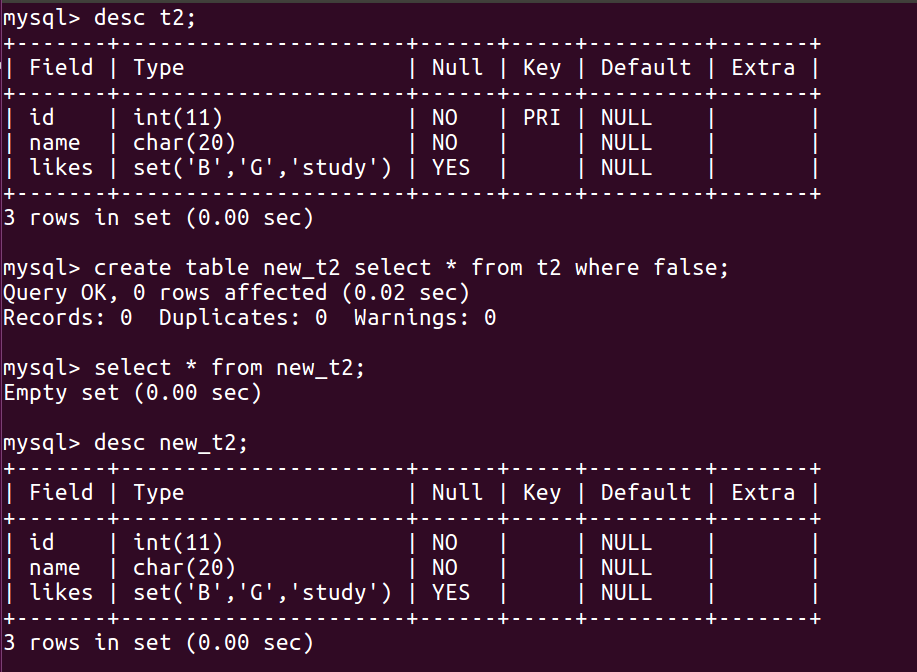
取名為userinfo4

(~ 後面加上limit 1,9)

2. 只複製表結構

1. 語法格式:

˙ create table 表名 select ... where false;



(見:MySQL03 copy structure)

＊注意

˙複製表的時候不會把原有表的"鍵屬性"複製過來

五、嵌套查詢

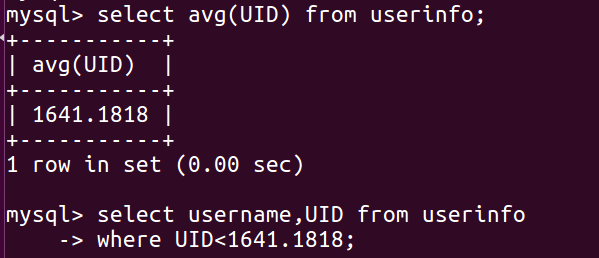
1. 定義: 把內層的查詢結果作為外層的查詢條件

2. 語法格式:

˙select 查詢語句 where 條件(select 查詢語句)

＊練習:

˙把UID值小於這個字段的平均值的用戶名和UID顯示出來



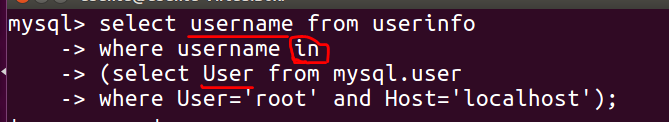
(作法1: MySQL03 nesting inquiry01圖) # 之前就會用的方式



(作法2: MySQL03 nesting inquiry02圖) # 嵌套的方式查詢

˙查找userinfo表中用戶名在mysql庫下的user表中的

Host字段值為localhost，並且User值是root的用戶名



(見:MySQL03 practice to Fourth圖)

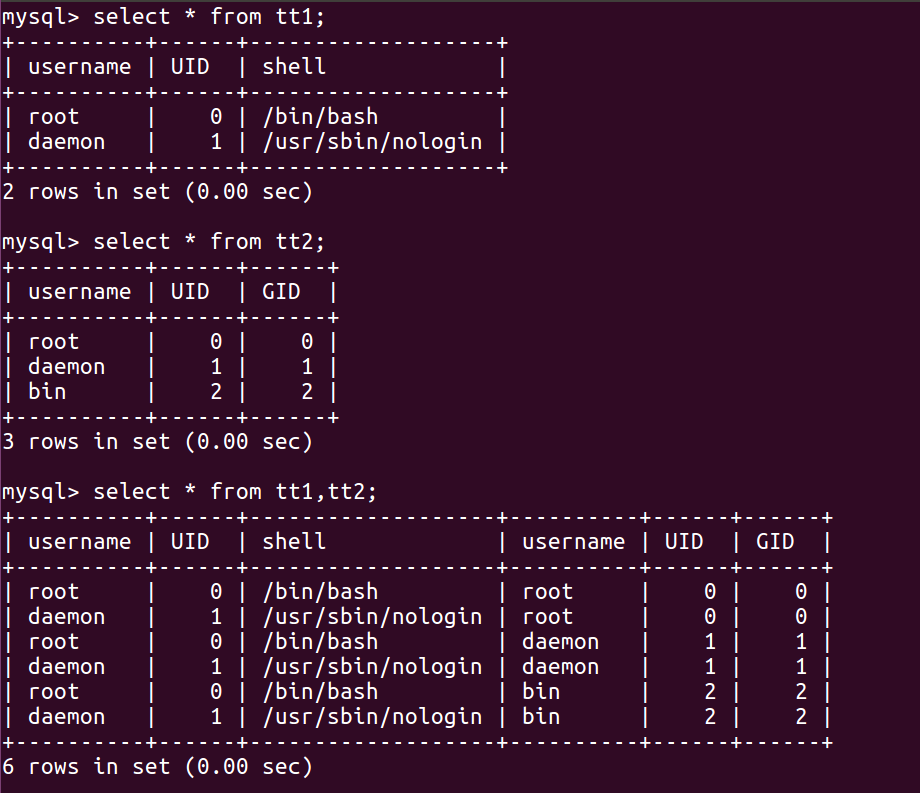
六、多表查詢

(表在db3的 sheng、city、xian裡面)

1. 兩種方式

˙select 字段名列表 from 表名列表; # 笛卡爾積

(以tt1、tt2表為例)



tt1的0,1匹配到tt2的0

(見:MySQL03 Multi-table query圖)

以tt1的所有紀錄和tt2的紀錄依序匹配

˙select 字段名列表 from 表名列表 where 條件

(查詢tt1和tt2表中 UID號相同的訊息)

select \* from tt1,tt2 where tt1.uid = tt2.uid;

＊練習:

1. 顯示省和市的詳細訊息

-> select sheng.S\_name,city.C\_name from sheng,city

-> where sheng.S\_ID = city.CFather\_ID;

2. 顯示省、市、縣的詳細訊息

> select sheng.S\_name , city.C\_name , xian.X\_name from

-> sheng,city,xian

-> where sheng.S\_ID = city.CFather\_ID

-> and city.C\_ID = xian.XFather\_ID;

七、連接查詢

1. 內連接

1. 定義: 從表中刪除與其他被連接表中沒有匹配到的紀錄

2. 語法格式:

˙select 字段名列表 from 表1 inner join 表2 on 條件;

＊練習

˙顯示省和市的詳細訊息，沒有匹配的不顯示

> select sheng.S\_name,city.C\_name from sheng

-> inner join city on sheng.S\_ID = city.CFather\_ID;

˙顯示省、市、縣的詳細訊息

> select sheng.S\_name,city.C\_name from sheng

-> inner join city on sheng.S\_ID = city.CFather\_ID

-> inner join xian on city.C\_ID = xian.XFather\_ID;

2. 外連接

1. 左連接

1. 定義: 以左表為主顯示查詢結果

2. 語法格式:

˙select 字段名列表 from 表1 left join 表2 on 條件;

＊練習

˙顯示省和市的詳細訊息，以左表為準(表1 為左表)

> select sheng.S\_name,city.C\_name from sheng

-> left join city on sheng.S\_ID = city.CFather\_ID;

＊與左表匹配不到的紀錄會以NULL呈現

2. 右連接

1. 定義: 以右表為主顯示查詢結果

2. 語法格式:

˙select 字段名列表 from 表1 right join 表2 on 條件;

＊練習

˙顯示省、市、縣的詳細訊息，要求市全部顯示

> select sheng.S\_name,city.C\_name,xian.X\_name from sheng

-> right join city on sheng.S\_ID = city.CFather\_ID

-> left join xian on city.C\_ID = xian.XFather\_ID;

(左連接與右連接的概念是相同的)

Day03 End.

MySQL進階IV:

Day 04:

一、數據備份(在linux終端操作)

1. 命令格式

˙mysqldump -u用戶名 -p[root密碼] 源庫名 > 路徑

(不見得要絕對路徑)

(如果不在p後面加密碼，則執行命令後也會要使用者輸入root密碼)

2. 源庫名的表示方式(使用者要備份的庫)

˙ --all-databases 備份所有庫

˙ 庫名 備份單個庫

˙-B 庫1 庫2 ... 備份多個庫

˙庫名 表1 表2 ... 備份指定庫的指定表

＊練習

˙備份所有庫為all\_mysql.sql放到主目錄下mydata目錄中

mysqldump -uroot -p --all-databases > ./all\_mysql.sql

˙備份三個庫，放到用戶主目錄下的mydata中

˙備份省、市、縣三張表

二、數據恢復(在linux終端操作)

1. 命令格式:

˙mysql -uroot -p 目標庫名(恢復到目標庫) < 路徑

(若恢復時沒有該目標庫，則要先創建一個空庫)

＊命令格式裡面沒有創建庫的語句

2. 從所有庫的備份文件中恢復某一個庫(--one-database)

˙mysql -uroot -p --one-database 目標庫名 <

(備份了所有庫的文件)

＊練習

˙先備份db3庫，刪除省、市、縣三張表

˙在db3庫中創建一張表 t888

create table t888(id int);

˙恢復db3庫

＊注意

1. 恢復庫時如果恢復到原有的庫會將表中數據覆蓋，

但新增的表不會刪除

2. 在恢復時如果要恢復的庫不存在，則要先創建空庫

三、ER模型 & ER圖

1. 定義: ER模型即 實體-關係 模型，ER圖即 實體-關係圖

2. 三個概念:

1. 實體

定義: 現實世界中可以被認知、區分的事物

舉例:

1. 學校: 學生、教師、課程、班主任 ...

2. 企業: 職員、產品 ...

2. 屬性

定義: 實體所具有的特性

舉例:

1. 學生屬性: 學號、姓名、班級、年齡、性別、專業 ...

2. 產品屬性: 產品編號、名稱、規格 ...

3. 關係

定義: 實體之間的關係

分類:

˙一對一的關係(1:1) 班級和班長

˙一對多的關係(1:n) 公司和職員、班級和學生

˙多對多的關係(m:n) 學生和課程、商店和顧客

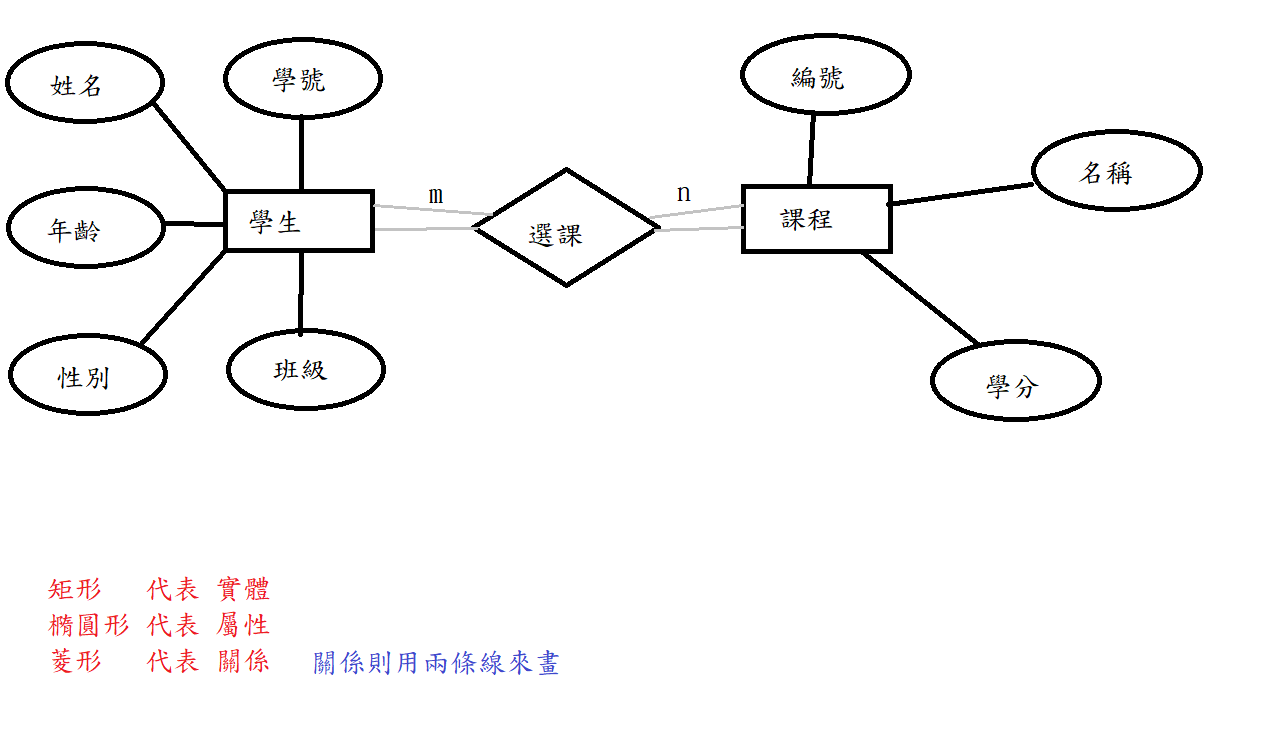
4. ER圖的繪製

˙矩形框 代表 實體

˙菱形框 代表 關係

˙橢圓形 代表 屬性

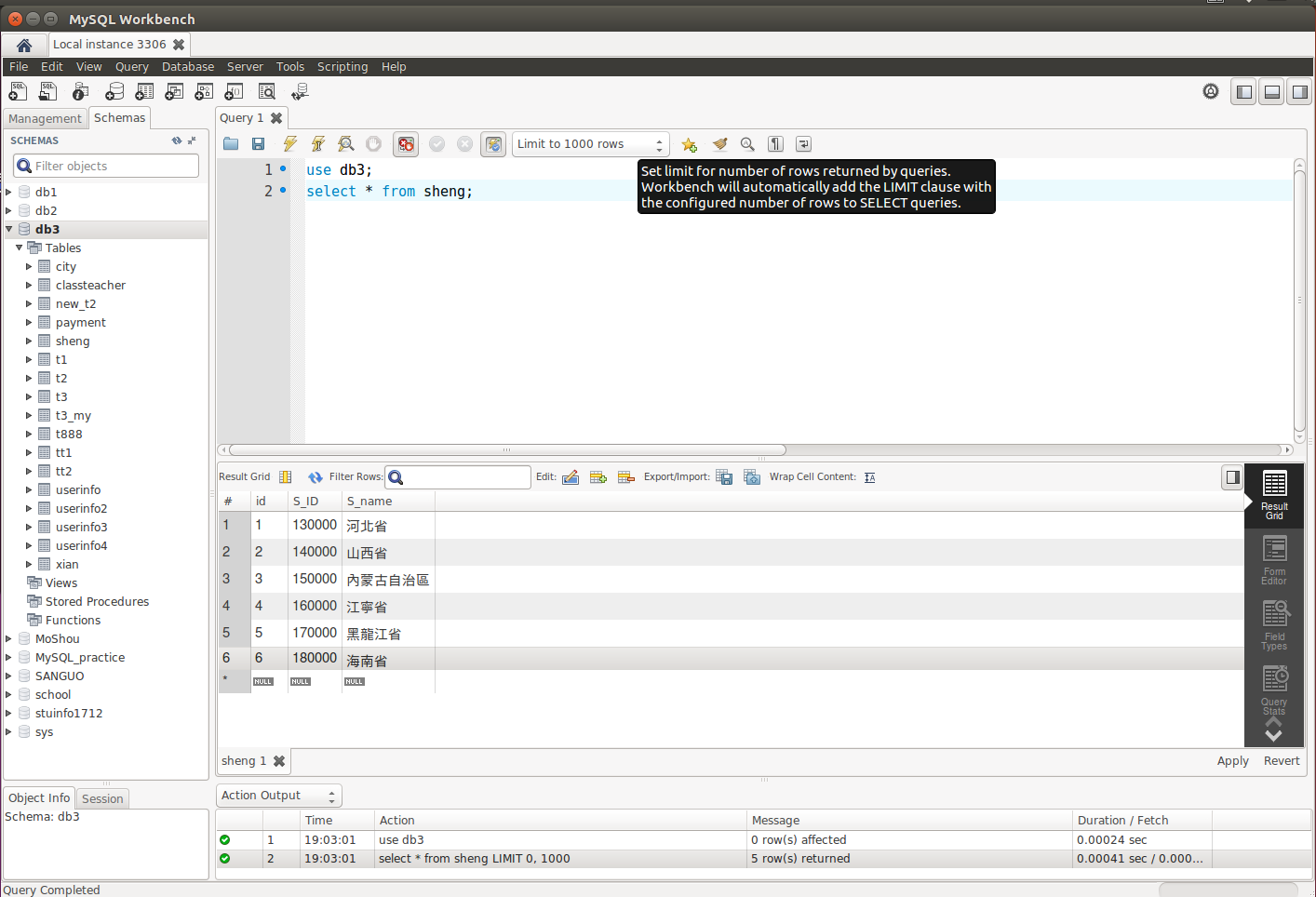
舉例:



學生選課系統(見:MySQL04 ER圖)

四、workbench使用(圖形化介面操作數據庫)

安裝 sudo apt-get install mysql-workbench -y



(見:MySQL04 mysql-workbench 操作介面圖)

建議不要太常使用這個應用

五、事務 & 事務回滾

1. 事務定義: 一件事從開始到結束的整個過程

2. 屬性

1. 原子性: 一個事務是不可分割的工作單位，事務中各個操作

要麼都做，要麼就都不做

2. 一致性: 事務必須從一個一致性狀態到另一個一致性狀態

3. 隔離性: 一個事務的執行不能被其他併發事務干擾

4. 持久性: 一個事務一旦提交，就會對數據庫造成永久性的改變

3. 事務及事務回滾的應用

1. mysql中默認SQL語句會自動提交(commit)到數據庫

˙show variables like "autocommit";

2. 事務應用

1. 開啟一個事務

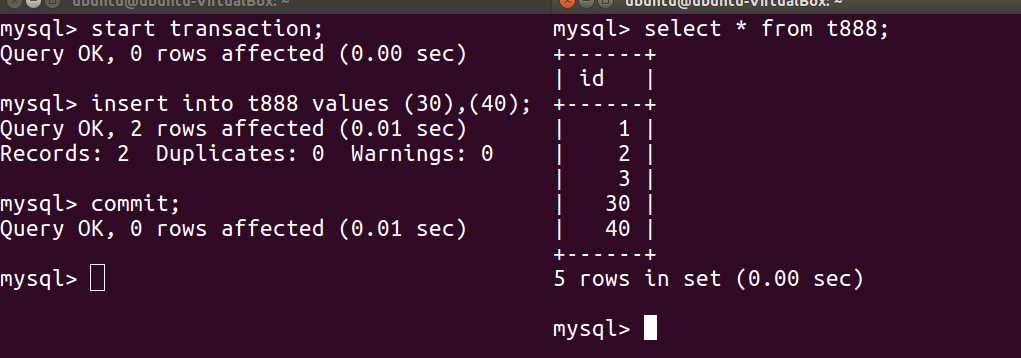
˙start transaction;

## 此時autocommit被禁用，SQL命令不會對數據庫中數據做修改

2. 終止事務

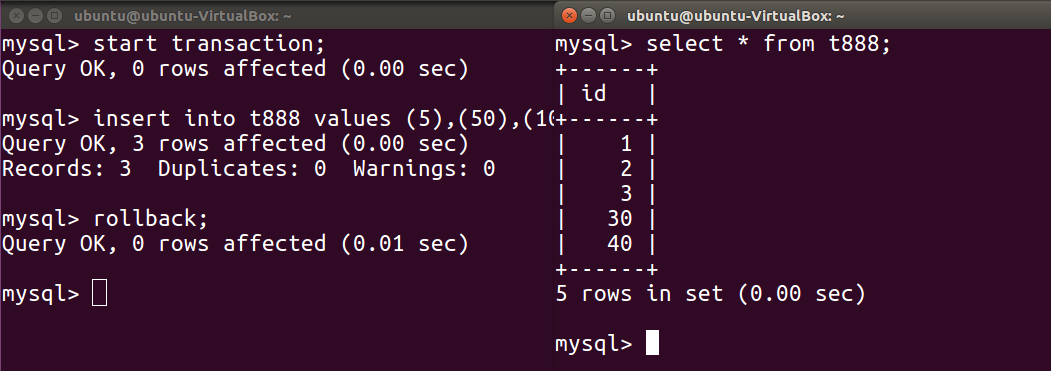
˙commit; | rollback; ## 提交 | 回滾

(提交則是會將 事務開始時到commit為止執行所有的語句命令)



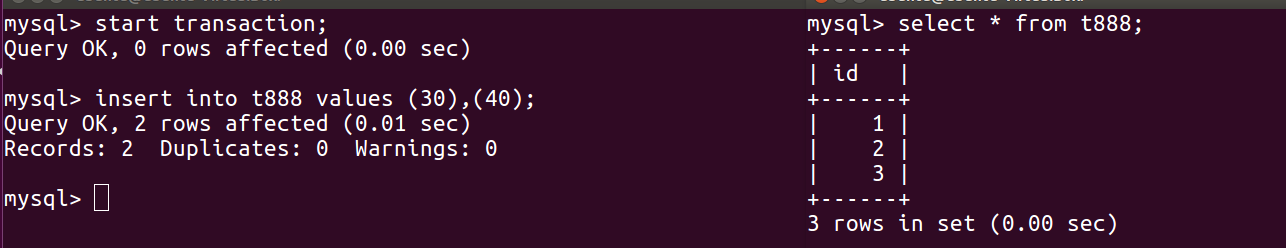
(見:MySQL04 transaction executed-commit圖)

(回滾則是會將 事物開始時到rollback為止的命令全不執行)



(見:MySQL04 transaction executed-rollback圖)

(若不提交也不回滾，即便有了指令也不會被執行)



(見:MySQL04 transaction executed圖)

＊注意

事務 & 事務回滾只針對對表紀錄的操作: 增加、刪除、修改

對創建庫、創建表是無效的

3. 案例

1. 背景

你 : 建行卡

你朋友 : 工商卡

你在建行的自動提款機給你朋友(工商銀行卡)轉帳

2. 過程

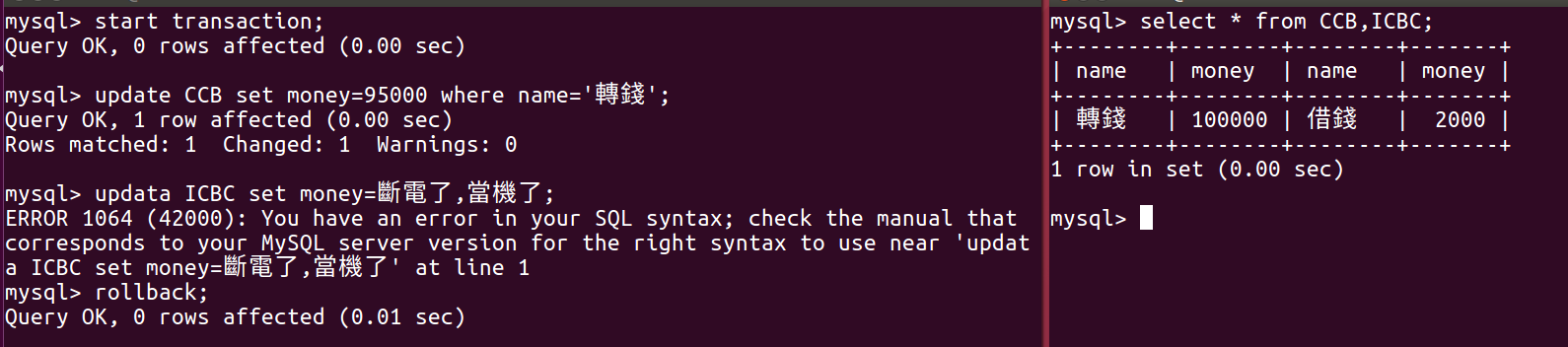
表1. 建行: CCB

表2. 工商: ICBC

-> 開始轉帳

工商銀行當機了

-> 轉帳失敗



(見:MySQL04 transaction 實例)

六、Python數據庫編程

1. Python數據庫接口(Python DB-API)

1. 定義: 為開發人員提供的數據庫應用編程接口

2. 支持的數據庫服務軟件

MySQL、Oracle、SQL\_Server、Sybase、Mogodb ...

3. Python 3提供操作MySQL的模塊

模塊名: pymysql

4. pymysql的使用流程

(以下6種方法)

˙建立數據庫連接

˙創建游標對象

˙使用游標對象的方法和SQL語句操控MySQL數據庫

˙提交commit

˙關閉游標

˙關閉數據庫連接

(見: FirstPymysql.py)

5. 建立數據庫連接

1. 語法格式:

˙對象名(db) = pymysql.connect(

"主機地址",'用戶名',"密碼","庫名"

,charset="utf8")

2. connect對象(db)的方法

˙db.cursor(): 創建一個游標對象db.cursor()

˙db.commit(): 提交到數據庫

˙db.rollback(): 回滾

˙db.close(): 關閉與數據庫的連接

3. cursor游標對象(cur)的方法

˙cur.execute(): 執行sql命令

˙cur.fetchone(): 取得結果集的第一條紀錄

˙fetchmany(數字n): 取得結果集到第 n 條紀錄

˙fetchall(): 取得結果集的所有行

˙close(): 關閉游標對象

屬性: rowcount: 傳回當前的數據條

pymysql.connect -> db -> db.cursor() -> 執行

(模塊:MysqlPython.py)

(見:Fetchone.py、Fetchmany-all.py、rollback.py、TextMysql.py)

FirstPymysql.py

import pymysql

# 打開數據庫連接(connect)

db = pymysql.connect("localhost",

"root",

"a123456",

charset="utf8")

# module.method("Host","User","Password",default character)

# 此時的charset=utf8 是指python內部執行器的默認字符集

# 創建一個游標(cursor)對象

cur = db.cursor()

# 創建庫python # exists 代表 存在

cur.execute("create database if not exists python;")

# 切換庫

cur.execute("use python;")

# 創建表t1 \為隱式換行

cur.execute("create table if not exists t1(\

id int primary key,\

name varchar(20),\

score tinyint unsigned)default charset=utf8;")

# 在t1表中插入5條紀錄

cur.execute("insert into t1 values\

(1,'Eric',80),(2,'Iris',90),\

(3,'Aurora',94),(4,'Lily',75),\

(5,'yuki',99);")

# 提交到數據庫

db.commit()

# 關閉游標

cur.close()

# 關閉數據庫連接

db.close()

Fetchone.py

import pymysql

db = pymysql.connect("localhost","root",

"a123456","python",charset="utf8")

cur = db.cursor()

sql\_select = "select \* from t1;"

cur.execute(sql\_select)

# fetchone 取得表中的第一條紀錄

data = cur.fetchone()

print("fetchone 的結果為:",data)

db.commit()

cur.close()

db.close()

Fetchmany-all.py

import pymysql

db = pymysql.connect("localhost","root",

"a123456","python",port=3306,charset="utf8")

# pymysql.connect(keyword argument)

#(Host,User,password[,purpose database][,port],charset)

cur = db.cursor()

sql\_select = "select \* from t1;"

cur.execute(sql\_select)

print("select 語句查出的紀錄個數為:",cur.rowcount)

# fetchmany(n) 取得結果集到第 n 條紀錄

data = cur.fetchmany(2)

print("fetchmany 的結果為:")

for i in data:

print(i)

data\_all = cur.fetchall()

print("\nfetchall 的結果為:")

for i in data\_all:

print(i)

db.commit()

cur.close()

db.close()

rollback.py

import pymysql as py

db = py.connect("localhost","root","a123456",

"db3",port=3306,charset="utf8")

cur = db.cursor()

try:

cur.execute("update CCB set money=95000 where name='轉錢';")

cur.execute("update ICBC set money=7000 where name='借錢';")

db.commit()

print("ok")

except Exception as e:

print(e)

db.rollback()

finally:

cur.close()

db.close()

MysqlPython.py

from pymysql import \*

class MysqlPython:

def \_\_init\_\_(self,host,user,password,db,

port=3306,charset='utf8'):

self.host = host

self.user = user

self.password = password

self.db = db

self.port = port

self.charset = charset

def open(self):

self.con = connect(host=self.host,user=self.user,

password=self.password,db=self.db,

port=self.port,charset=self.charset)

self.cursor = self.con.cursor()

def close(self):

self.cursor.close()

self.con.close()

def exe(self,sql):

self.open()

self.cursor.execute(sql)

self.con.commit()

self.close()

def all(self,sql):

try:

self.open()

self.cursor.execute(sql)

data = self.cursor.fetchall()

return data

except Exception as e:

print(e)

finally:

self.close()

TextMysql.py

from MysqlPython import MysqlPython

# update

name = input("please input modify student name:")

score = int(input("please input the student new score:"))

sql = "update t1 set score='%s' where name='%s';"% (score,name)

sqlH = MysqlPython("localhost","root","a123456",

"python")

sqlH.exe(sql)

d = sqlH.all("select \* from t1;")

for i in d:

print(i)

Day04 End.