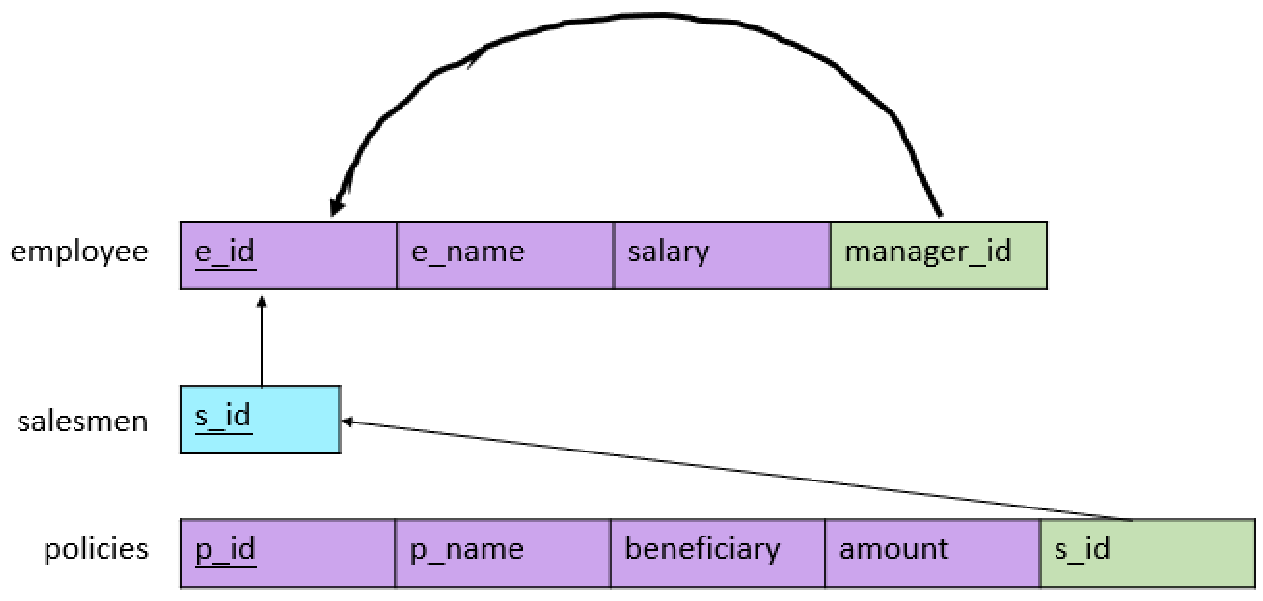
HW2B Report

409410050 資工二 王謙靜

part6

relational schema



建立table之DDL

employee

```
CREATE TABLE employee (
    e_id int PRIMARY KEY NOT NULL,
    e_name varchar(30) NOT NULL,
    salary varchar(30) NOT NULL,
    manager_id int,
    CONSTRAINT manage_by_fk FOREIGN KEY (manager_id) REFERENCES employee(e_id)
);
```

salesman

```
CREATE TABLE salesmen(
    s_id int PRIMARY KEY NOT NULL,
    CONSTRAINT salesmen_fk FOREIGN KEY (s_id) REFERENCES employee(e_id)
);
```

policies

```
CREATE TABLE policies (
    p_id int PRIMARY KEY NOT NULL,
    p_name varchar(30) NOT NULL,
    beneficiary varchar(30) NOT NULL,
    amount int NOT NULL,
    s_id int NOT NULL,
    CONSTRAINT sold_fk FOREIGN KEY (s_id) REFERENCES salesmen(s_id)
);
```

假設

1. 會查詢salesmen有誰
2. 每個員工都可以找到至多一一個manager

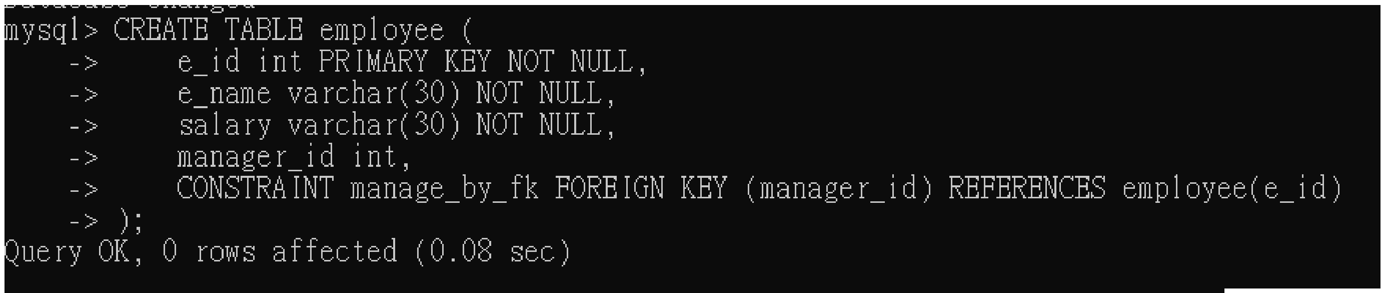
entity

employee:有三個attritube，且key是ID，故employee的table中有三個column: ID, Name, Salary。
policies:有四個attritube，且P#是key，故policies的table中有四個column:p_id, p_name, beneficiary, amount。(由於無法以'#'命名，故將P#改名為p_id)
salesmen:繼承自employee，故其key為employee，因為可能會有查詢問說誰是salesmen，若不保留則無法得知。salesmen table中有一個column:s_id，且s_id是employee中e_id的foreign key，其有constraint限制每個salesmen都必須是employee。

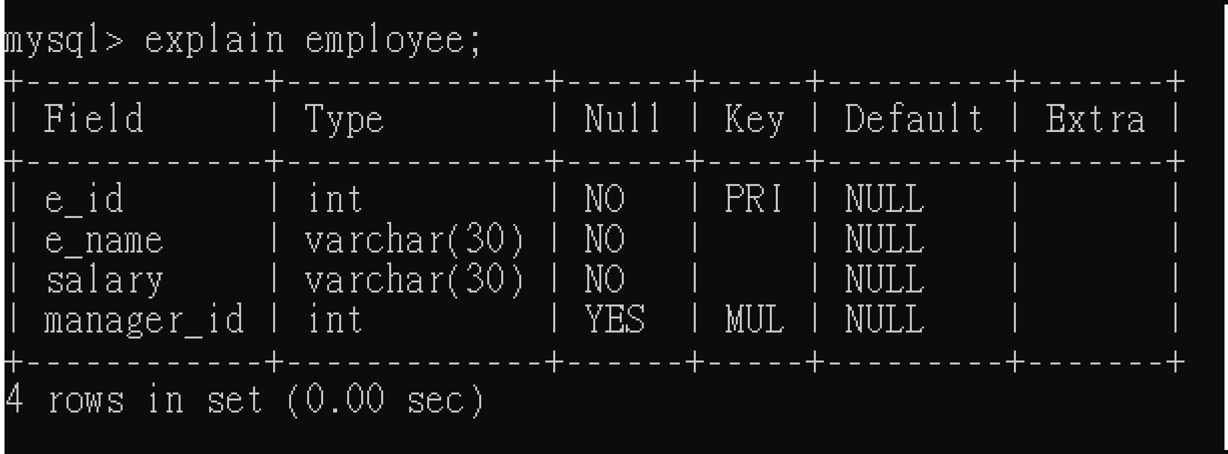
relation

manage:是一個一對多的關係，每個員工都可以找到至多一一個manager，故在employee中有另一column :manager_id，紀錄每個employee的manager，其有Foreign key指向employee，代表manager一定要是employee。
sold:是一對多的關係，每個policies都可以對應到唯一一個salesmen，故在policies中另有一個column: s_id，記錄其對應到的salesmen，其有Foreign key指向salesmen，檢查s_id是否在salesmen中。由於policies與sold之間有total participation所以s_id保證為空。

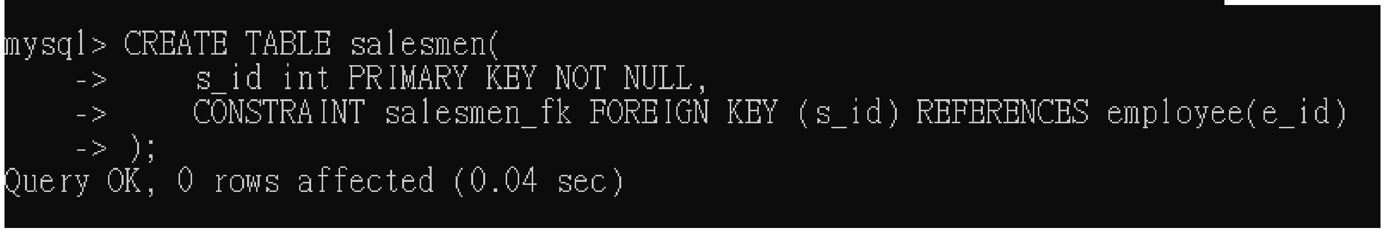
截圖



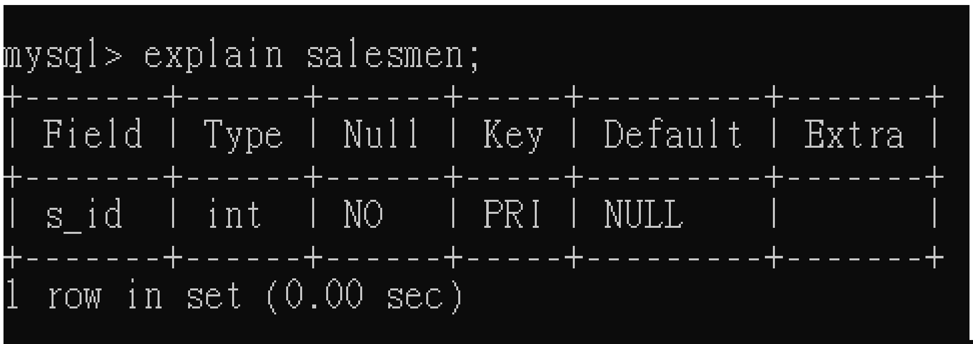
```
mysql> CREATE TABLE employee (
    ->     e_id int PRIMARY KEY NOT NULL,
    ->     e_name varchar(30) NOT NULL,
    ->     salary varchar(30) NOT NULL,
    ->     manager_id int,
    ->     CONSTRAINT manage_by_fk FOREIGN KEY (manager_id) REFERENCES employee(e_id)
    -> );
Query OK, 0 rows affected (0.08 sec)
```



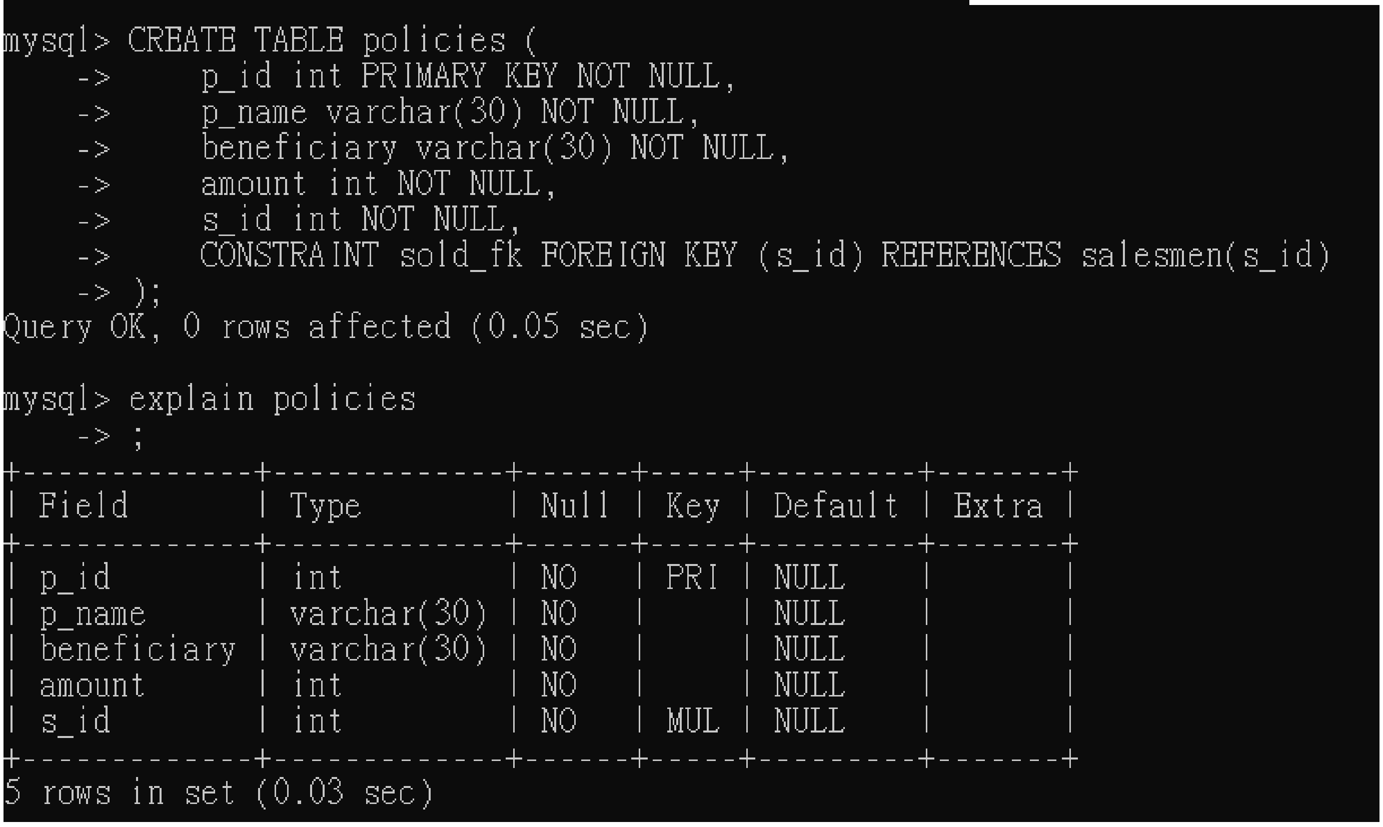
```
mysql> explain employee;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| e_id       | int         | NO   | PRI | NULL    |       |
| e_name     | varchar(30) | NO   |     | NULL    |       |
| salary     | varchar(30) | NO   |     | NULL    |       |
| manager_id | int         | YES  | MUL | NULL    |       |
+------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```



```
mysql> CREATE TABLE salesmen(
    ->     s_id int PRIMARY KEY NOT NULL,
    ->     CONSTRAINT salesmen_fk FOREIGN KEY (s_id) REFERENCES employee(e_id)
    -> );
Query OK, 0 rows affected (0.04 sec)
```



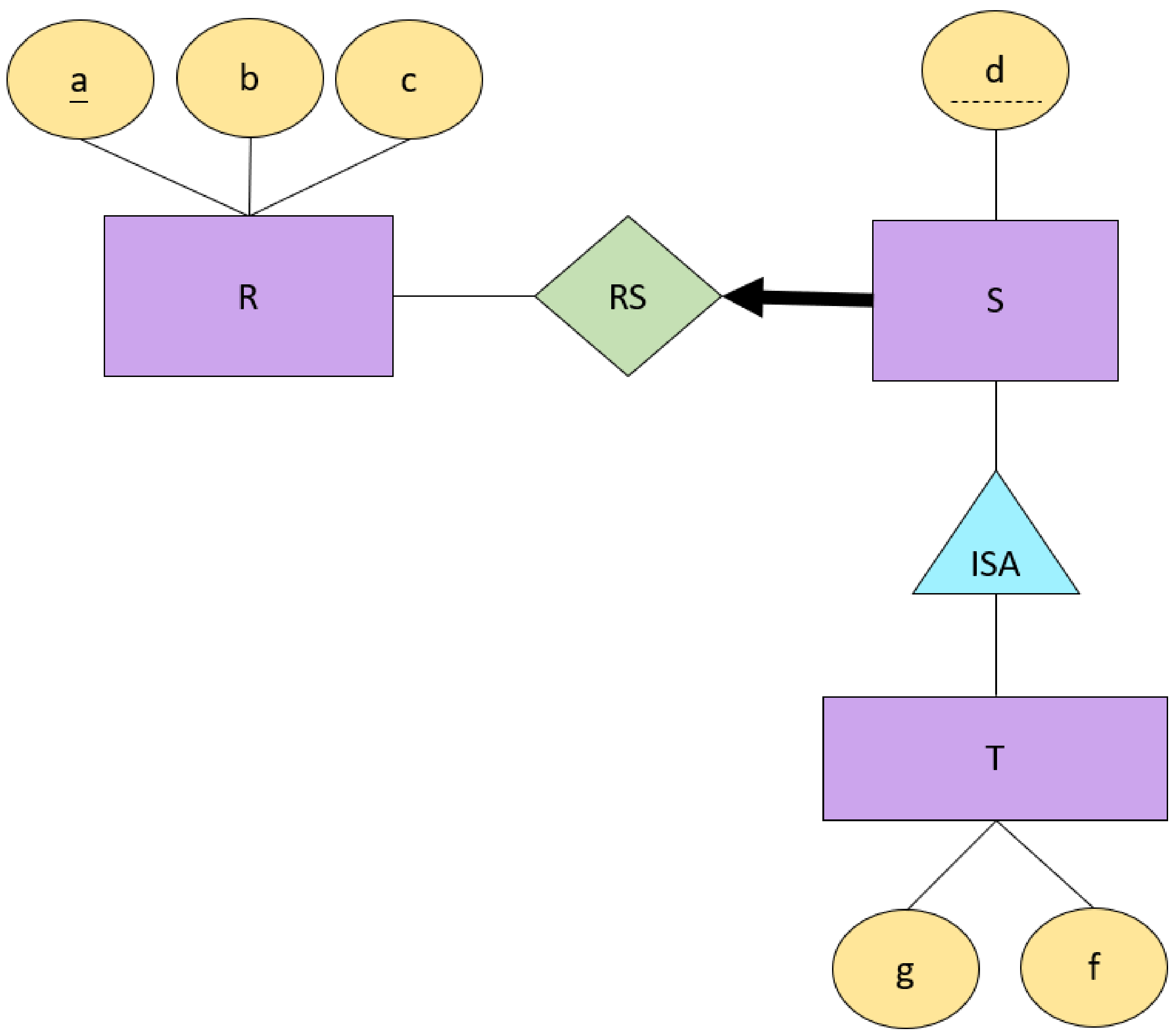
```
mysql> explain salesmen;
+-------+------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+-------+------+------+-----+---------+-------+
| s_id  | int  | NO   | PRI | NULL    |       |
+-------+------+------+-----+---------+-------+
1 row in set (0.00 sec)
```



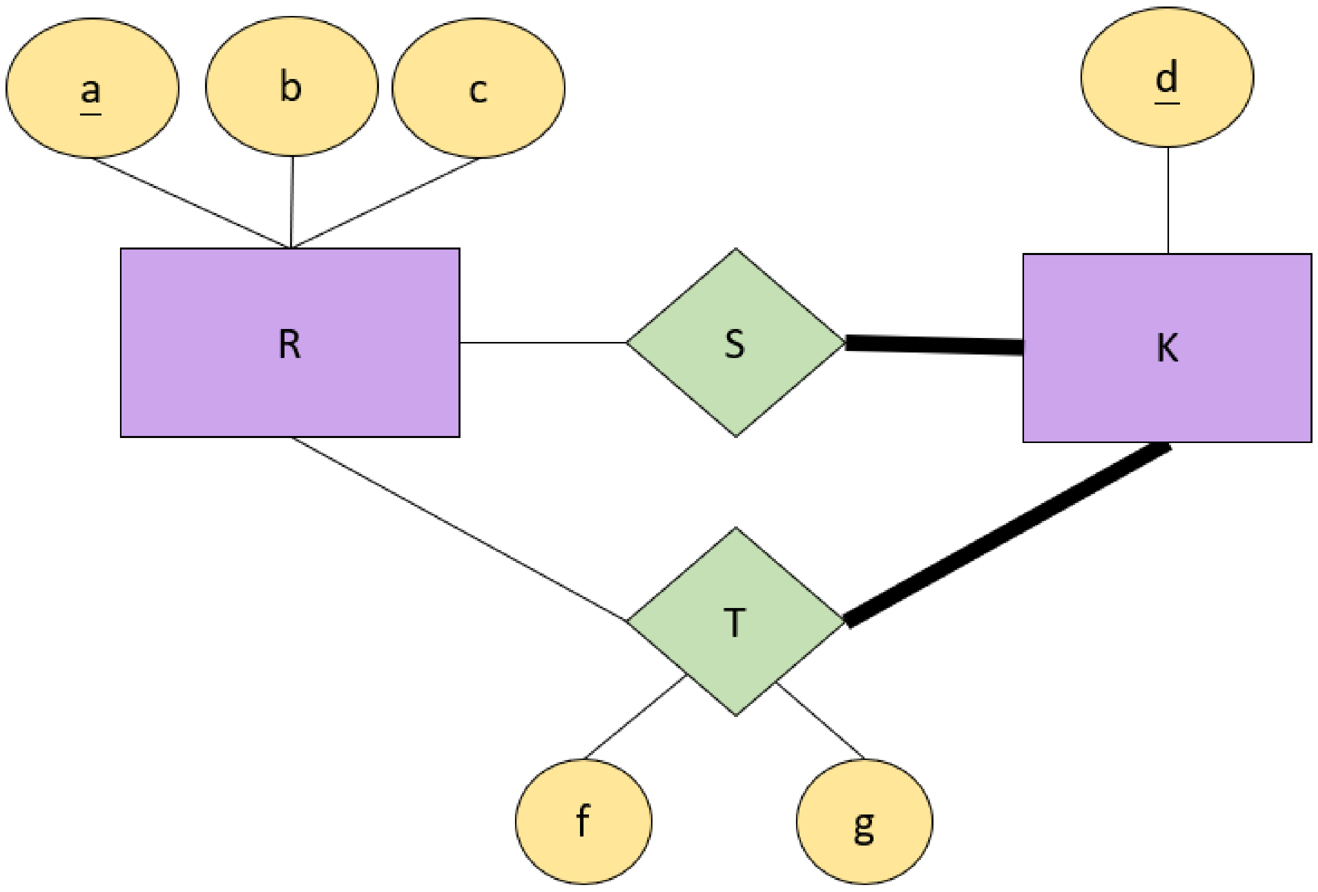
```
mysql> CREATE TABLE policies (
    ->     p_id int PRIMARY KEY NOT NULL,
    ->     p_name varchar(30) NOT NULL,
    ->     beneficiary varchar(30) NOT NULL,
    ->     amount int NOT NULL,
    ->     s_id int NOT NULL,
    ->     CONSTRAINT sold_fk FOREIGN KEY (s_id) REFERENCES salesmen(s_id)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> explain policies
    -> ;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| p_id        | int         | NO   | PRI | NULL    |       |
| p_name      | varchar(30) | NO   |     | NULL    |       |
| beneficiary | varchar(30) | NO   |     | NULL    |       |
| amount      | int         | NO   |     | NULL    |       |
| s_id        | int         | NO   | MUL | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
5 rows in set (0.03 sec)
```

part7

圖一



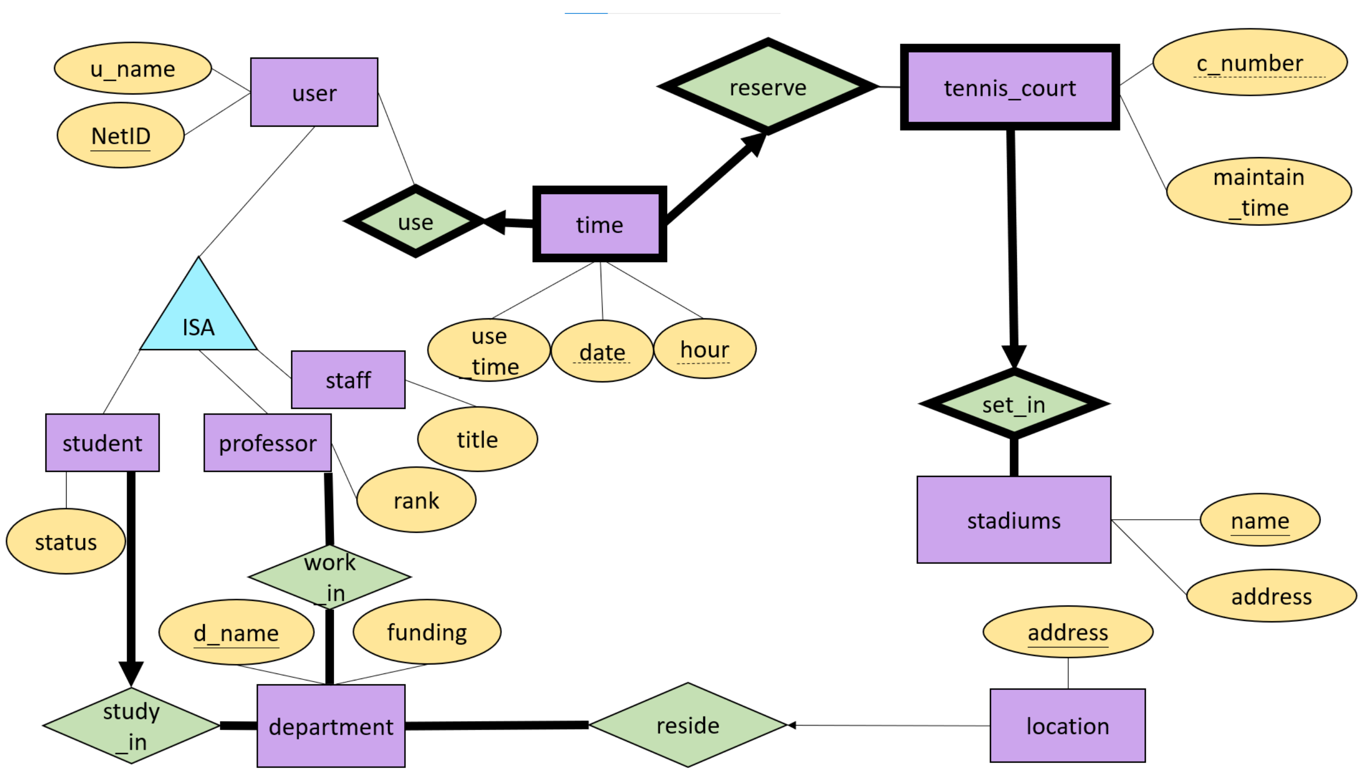
R：R有三個attritube，所以R table中有三個column: a, b, c，其中a為key。
S：S是一個weak entity，所以要把owner entity的key也寫進S的table中，故S table中有兩個column:a, d，其中a, d合起來為一個key。
T：T繼承於S。故要把S的key也寫進T的table中，另有兩個attritube，故T table有四個column:a, d, f, g，其中a, d為key。
圖二



R：R有三個attritube，所以R table中有三個column: <u>a</u>, b, c，其中a為key。
S：由於S relation兩邊為多對多，所以S table有兩個column: a, d，且以a, d兩者合併當作key
T：由於T relation兩邊為多對多，所以T table有兩個column: a, d，且以a, d兩者合併當作key。而T另有兩attritube，故T table另有兩個column:f, g
K:由於k沒有其他的attritube，且所有k都會參加S relation，所以可以不用用另一個table紀錄，想知道k有誰有可以查詢所有S table的所有d。

part8

ER-diagram



假設

1. 每個位置(location)只會有一個科系(department)，且每個科系都要有位置。
2. 對於每個科系(department)需要另外記錄其所擁有的經費
3. 要記錄每個網球場上一次的維修時間(remain_time)
4. time slot中要記錄使用者的使用時間(use_time)
5. 每個學生都必定恰好有一個科系
6. 每個科系都一定會有教授和學生，且可能不只一個
7. 每個體育館都有球場

entity

user：透過 `A user is identified by the NetID. We also record the name of the user.` 的文字說明，我們可以知道有一個user entity，並有NetID作為key attritube以及u_name做non key attritube
student：透過 `We have three kinds of users: student, professor, and staff. Students additionally have a status recorded (such as freshman, sophomore, junior, senior, MS, PhD)` 的說明，我們可以知道student其實是user的一種，因此student繼承了user(在圖上用ISA表示)，並另外有一個atrritube: status紀錄目前學生的年級。
professor：透過 `We have three kinds of users: student, professor, and staff.` 的說明，我們可以知道professor繼承了user，故在圖上用ISA表示。透過 `A professor additionally has a record of the rank (assistant professor, associate, etc.).` ，可以知道另外有一個atrritube: rank紀錄目前教授的等級(教授、副教授...等)。
staff：透過 `We have three kinds of users: student, professor, and staff.` 的文字說明，我們可以知道staff繼承了user，故在圖上用ISA表示。透過 `We also record the title of a staff` 的說明，可以知道staff要記錄其職稱，故另外有一個atrritube: title。

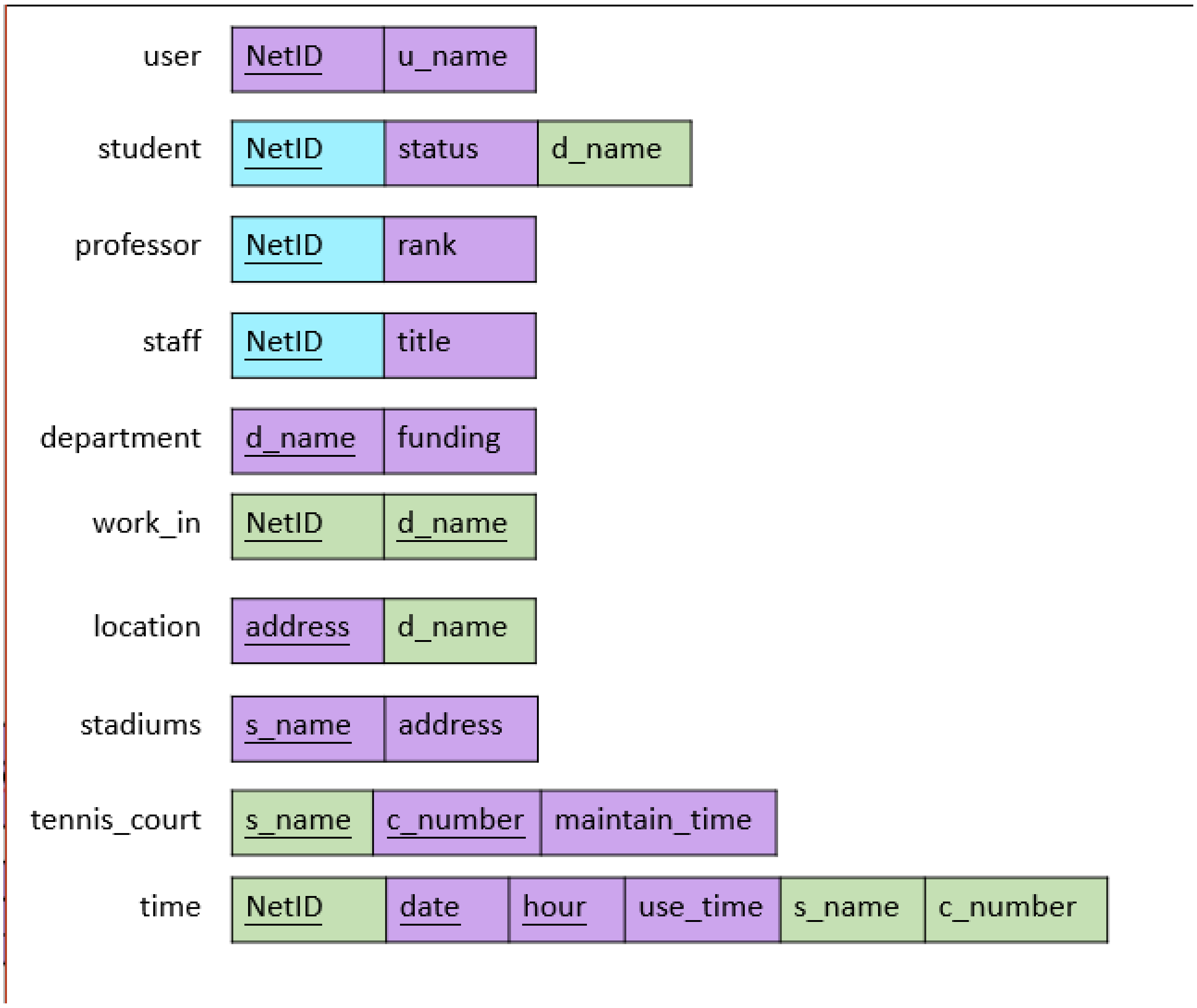
department：透過 `There are departments in the university.` 以及 `A professor can be affiliated with many departments.` 可以知道需要紀錄department，且因professor可能有多個department，故department不能作為attritube存在professor的entity中。透過 `A department is identified by its name.` 以及假設2，可以知道department有一個key attritube: d_name以及non key attritube: funding。
location：透過 `A location is identified by its address. A department may reside in multiple locations.` 由於一個department可能有多個location，所以不能夠以attritube的形式存在於ER-diagram中，故讓location成為一個entity並有address當作key。
stadiums：`The stadiums體育場 have unique names and can be at one location.` 得知需要一個stadiums entity，並以s_name當key，且另有一個名為location的attritube。
tennis_court： `A tennis court is identified by the name of the stadium that contains it, and its own court number.` 可知有tennise_court entity是由stadiums的key以及自己的partial key: number合併當作key的entity，由於需要其他entity才能確定唯一，故tennis_court是weak entity，其identity relation是set_in(set_in為記錄球場與體育館的relation)。由於假設3，另外有一個atrritube:maintain_time。
time： `We keep track of reservation time slots by its date and hour.` 可知有一個entity叫作time，他有date，和hour(分別表示日期與時間)，由於假設4之下，不同的date與hour可能會有不同的使用時間，所以time是一個weak entity。透過user或者tennise_court都可以唯一決定一個時間資料，故user與tennis_court都是time的owner entity，透過NetID或者tennis_court的key加上date和hour都可以唯一找到一組time，故time的identity relation可以是use或revserve。

relation

study_int：記錄學生與科系之間的關係，透過 `A student can only belong to one department.` 可以知道學生有唯一的科系且必定要有一個科系，故student與study_in之間有key constraint和total participation。因為假設每個科系都會有學生，所以department與study_in間有total participation，但因科系可能有不只一個學生，所以沒有key constraint。
work_in：記錄教授與科系之間的關係，透過 `A professor can be affiliated with many departments.` 可知教授與科系為多對多的relation，可能有一個教授屬於多個科系，也可能有一個科系有多個教授，所以professor與work_in間和department與work_in之間都沒有key constraint。因為每個教授都必須要有科系，且每個科系都要有教授，所以professor與work_in間和department與work_in之間都有total participation。
reside：記錄每個科系的位置，由於一個系可能有多個位置，且每個科系都要有至少一個位置，故department與reside間有total participation但沒有key constraint。因假設一個位置只會有一個科系，且不一定所有位置都是系館，所以location與reside之間只會有key constraint而沒有total participation。
set_in：網球場與體育館的關係(網球場設立在體育館中)，因為網球場可以靠它找到唯一一個體育館，並且所有網球場都隸屬於體育館內，故set_in與tennis_court之間有total participation與key constraint。由於tennis_court必須透過set_in來找到一個體育館使自己是unique，故set_in是一個identity relation(relation 加粗)。由於每個體育館可能有多個球場，且每個體育館都要有球場，所以set_in與stadiums之間沒有key constraint，但有total participation
use：記錄使用者與time的關係，因為每個使用者可能在不同時間使用球場，且非每個使用者都

會使用球場(可能有人不會打網球)，故user與use之間沒有total participation以及key constraint。因為每一筆time都必須洽好屬於一個user，所以time與use之間有total participation以及key constraint。由於time可以透過use以及自己的partial key來唯一決定一筆資料，所以use是time的identity relation(use 加粗)。
reserve : 記錄time與球場的關係，因為每個球場可能在不同時間被使用，且非每個球場都會被使用(可能特別遠沒有人想用)，故tennis_court與reserve之間沒有total participation以及key constraint。因為每個預約的時間都必須恰好有一個球場，所以time與reserve之間有total participation以及key constraint。由於time可以透過tennis_court以及自己的partial key來唯一決定一筆資料，所以reserve是time的identity relation。

relational schema



user	NetID	u_name				
student	NetID	status	d_name			
professor	NetID	rank				
staff	NetID	title				
department	d_name	funding				
work_in	NetID	d_name				
location	address	d_name				
stadiums	s_name	address				
tennis_court	s_name	c_number	maintain_time			
time	NetID	date	hour	use_time	s_name	c_number

user, **department**, **stadiums**, **location**: 因為他是strong entity，所以直接將它變成table，且把ER-diagram上的key當成table的key。
student, **professor**, **staff**:三者都繼承自user，把user的key當作table的key並加入對應各自atrritube的column。由於user一定要存在，才可能有student, professor, staff，所以NetID是Foreign key，必須要constraint才能確保student, professor, staff的合理性。
study_in:他是一個一對多的關係，每個學生都只會有一個科系，所以可以在student的table新增一個column記錄department的key。由於有total participation所以新增的column的值不會有NULL。d_name也是forign key，因此也需有constraint確定科系存在

work_in:多對多的關係，故將它畫成一個table，並把professor的key和department的key寫進table中，並把兩個column綁在一起當key。NetID與d_name都是foreign key，NetID是存professor，必須確定professor合法，才能有work_in關係，d_name則是存科系，同樣要確定科系合法，才能有work_in關係。
reside:一(department)對多(location)的relation，所以將deparment的值寫進location的另一個column中，由於location與reside之間沒有total participation，所以會有NULL。由於把department的key寫進reside，故新增的column也是foreign key。
tennis_court, **set_in**:他是一個weak entity，他的identity relation是set_in，owner entity是stadiums，所以把stadiums的key以及自己的partial key合在一起當key，並將另一個attritube轉換成column。其中s_name是foreign，他來自stadiums。
time, **use**, **reserve**:time是一個weak entity，user的key與time的partial key和tennis_court的key與time的partial key合起來都可以是time的key(use與reserve皆為identity relation)，在這裡我們選擇用user作為owner entity，因為key的數量較少，所以選擇user當owner entity。將user的key:NetID和time的partial key綁在一起，time的schema中有NetID, date, hour三個組合而成key，以及一個use_time column(attritube直接轉成column)。而reserve relation則另外處理，由於reserve是一(time)對多(tennis_court)的關係，所以我們可以在time table中新增column，並將tennis_court的key存在那個column中，由於有total participation，所以那個column為非空。其中NetID(來自user), s_name, c_number(來自tennis_court)為foreign key。

DDL

user

```
CREATE TABLE user (
    NetID int PRIMARY KEY NOT NULL,
    u_name varchar(30) NOT NULL
);
```

department

```
CREATE TABLE department (
    d_name varchar(30) PRIMARY KEY NOT NULL,
    funding int NOT NULL
);
```

student

```
CREATE TABLE student (
    NetID int PRIMARY KEY NOT NULL,
    status varchar(30) NOT NULL,
    d_name varchar(30) NOT NULL,
    CONSTRAINT user_student_fk FOREIGN KEY (NetID) REFERENCES user(NetID),
    CONSTRAINT depart_student_fk FOREIGN KEY (d_name)
    REFERENCES department(d_name)
);
```

professor

```
CREATE TABLE professor (
    NetID int PRIMARY KEY NOT NULL,
    p_rank varchar(30) NOT NULL,
    CONSTRAINT user_professor_fk FOREIGN KEY (NetID) REFERENCES user(NetID)
);
```

staff

```
CREATE TABLE staff (
    NetID int PRIMARY KEY NOT NULL,
    title varchar(30) NOT NULL,
    CONSTRAINT user_staff_fk FOREIGN KEY (NetID) REFERENCES user(NetID)
);
```

work_in

```
CREATE TABLE work_in(
    NetID int NOT NULL,
    d_name varchar(30)NOT NULL,
    PRIMARY KEY (NetId, d_name),
    FOREIGN KEY (NetID) REFERENCES professor(NetID),
    FOREIGN KEY (d_name) REFERENCES department(d_name)
);
```

location

```
CREATE TABLE location(
    address varchar(30) PRIMARY KEY NOT NULL,
    d_name varchar(30) NOT NULL,
    CONSTRAINT locate_fk FOREIGN KEY (d_name) REFERENCES department(d_name)
)
```

stadiums

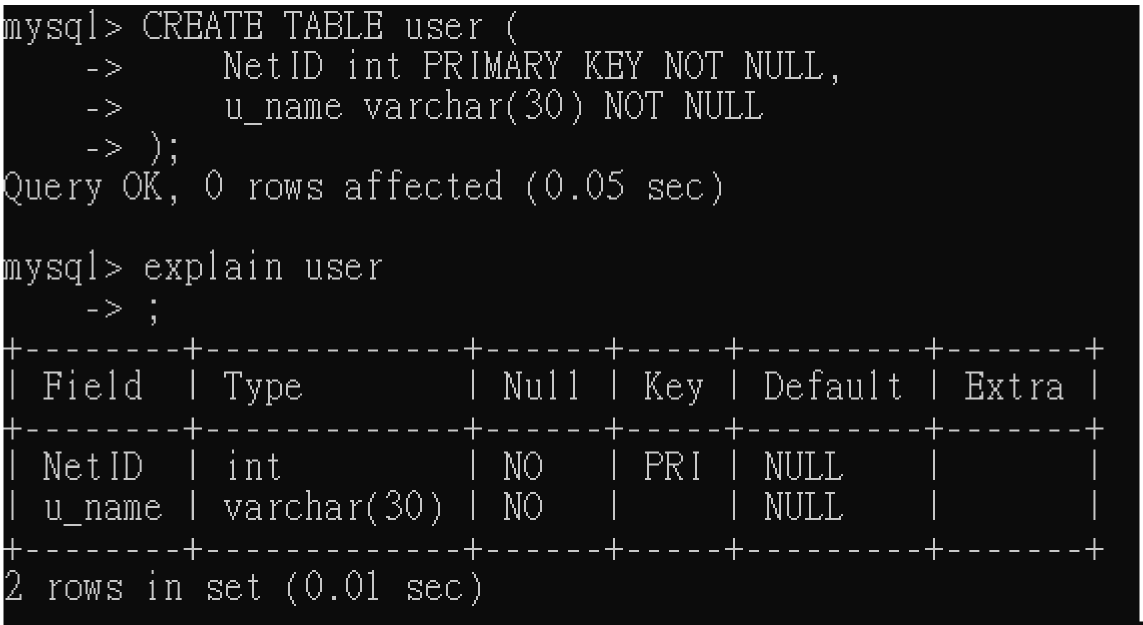
```
CREATE TABLE stadiums(
    s_name varchar(30) PRIMARY KEY NOT NULL,
    address varchar(30) NOT NULL
)
```

tennis_court

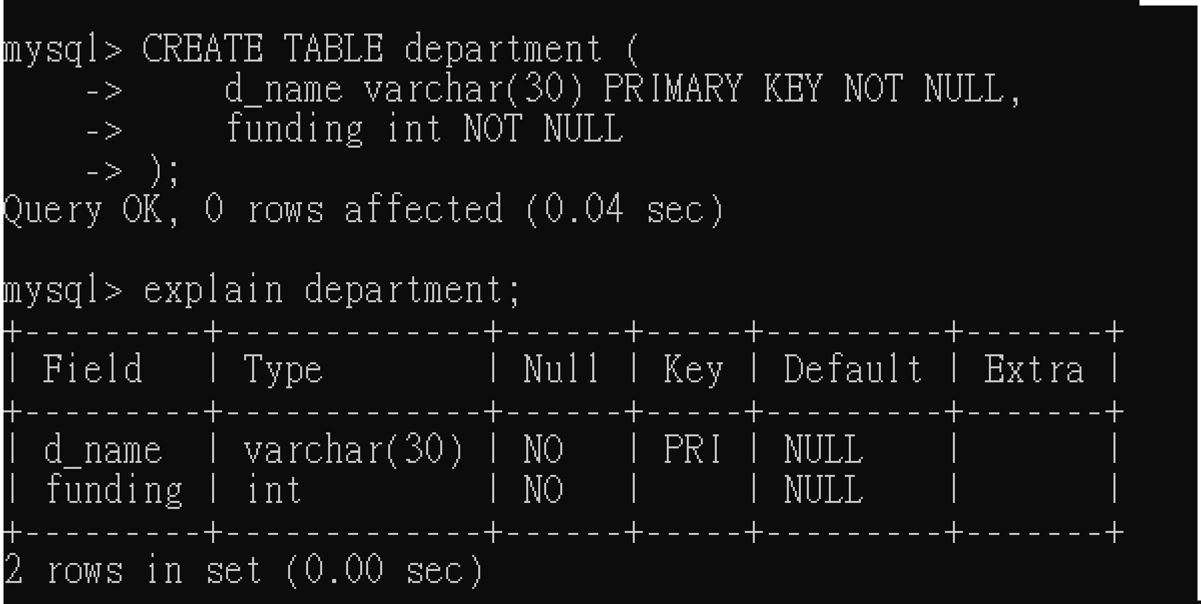
```
CREATE TABLE tennis_court(
    s_name varchar(30)NOT NULL,
    c_number int NOT NULL,
    maintain_time varchar(30) NOT NULL,
    PRIMARY KEY (s_name, c_number),
    CONSTRAINT stadiums_fk FOREIGN KEY (s_name) REFERENCES stadiums(s_name)
)
```

time

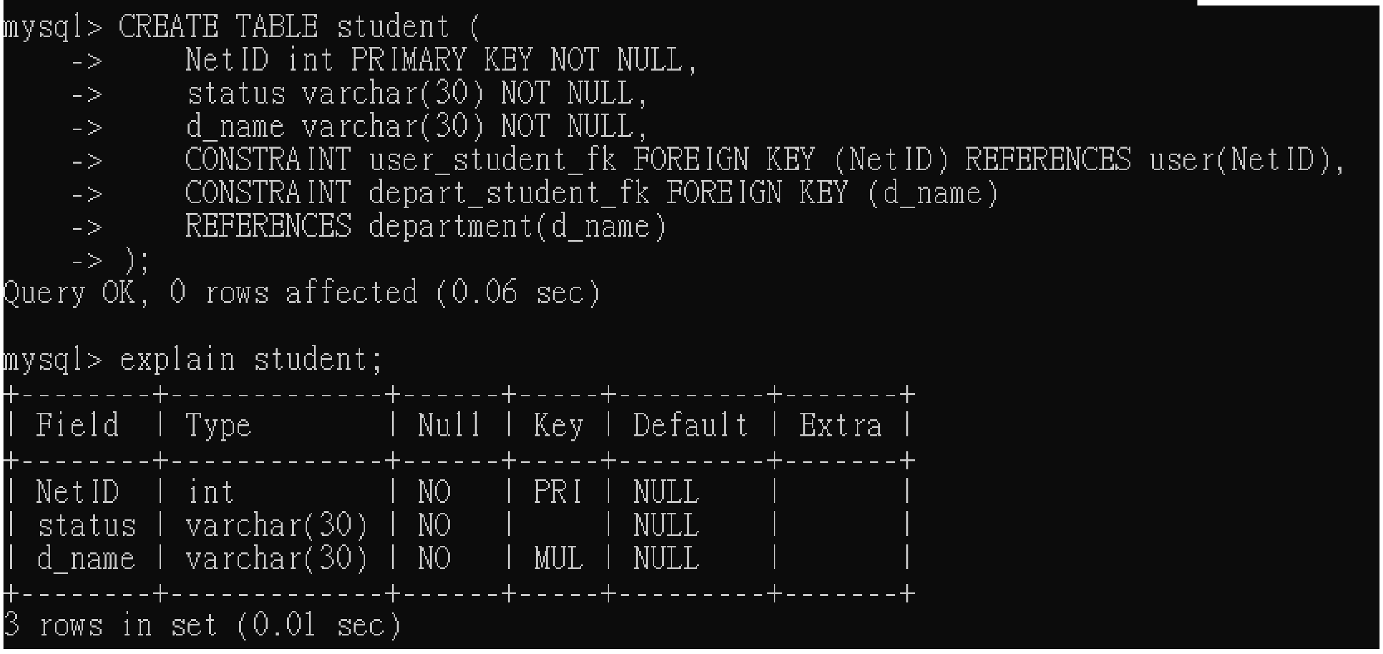
```
CREATE TABLE time(
    NetID int  NOT NULL,
    date varchar(20) NOT NULL,
    hour int NOT NULL,
    use_time int NOT NULL,
    s_name varchar(30) NOT NULL,
    c_number int  NOT NULL,
    PRIMARY KEY (NetID, date, hour),
    UNIQUE KEY(s_name, c_number, date, hour),
    CONSTRAINT ti_stadiums_fk FOREIGN KEY (s_name, c_number)
    REFERENCES tennis_court(s_name, c_number),
    CONSTRAINT user_ti_fk FOREIGN KEY (NetID)
    REFERENCES user(NetID)
)
```



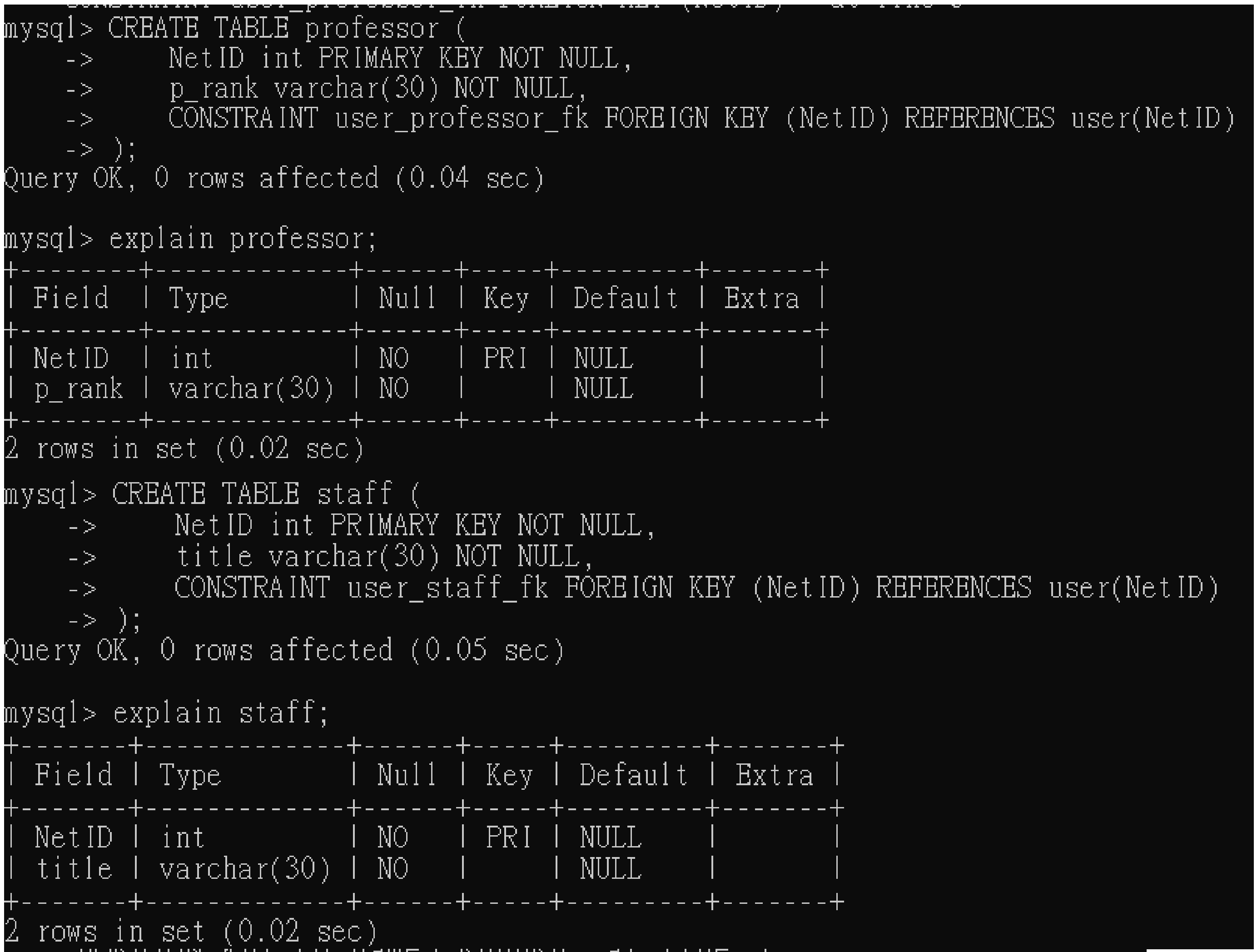
```
mysql> CREATE TABLE user (
    ->     NetID int PRIMARY KEY NOT NULL,
    ->     u_name varchar(30) NOT NULL
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> explain user
    -> ;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| NetID  | int         | NO   | PRI | NULL    |       |
| u_name | varchar(30) | NO   |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
2 rows in set (0.01 sec)
```



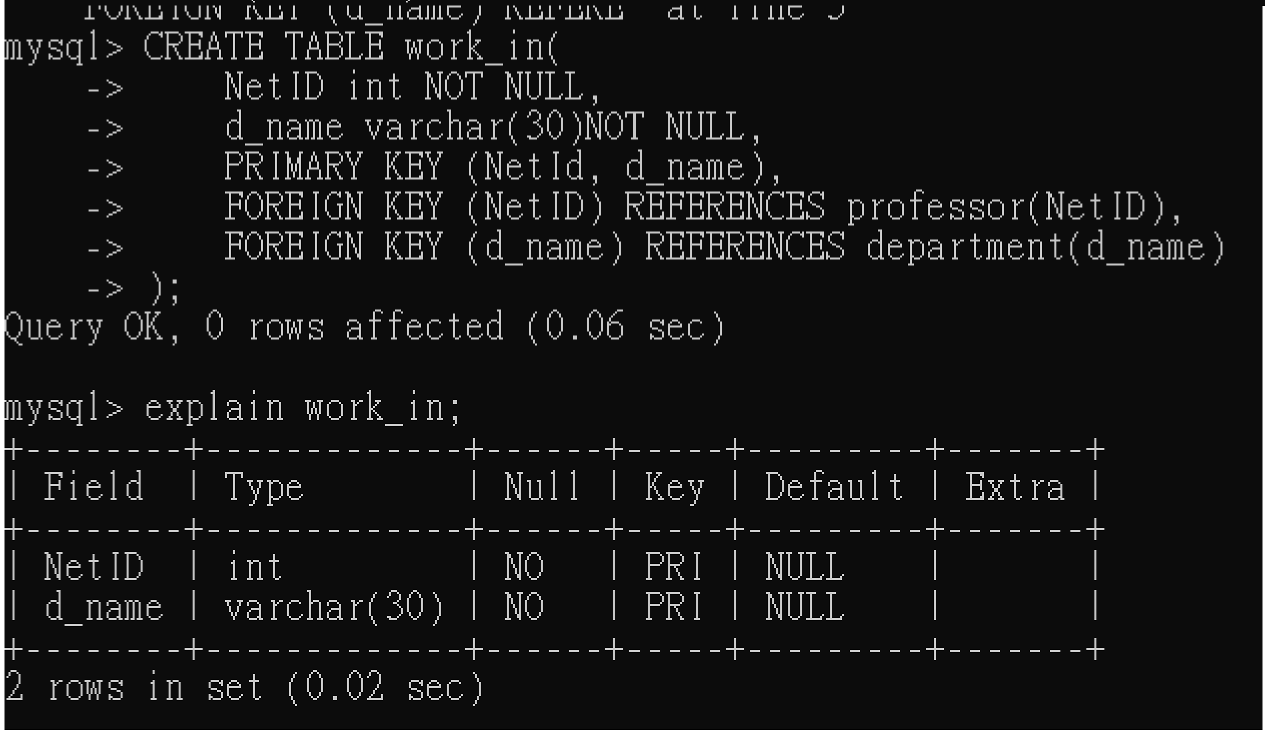
```
mysql> CREATE TABLE department (
    ->     d_name varchar(30) PRIMARY KEY NOT NULL,
    ->     funding int NOT NULL
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> explain department;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| d_name  | varchar(30) | NO   | PRI | NULL    |       |
| funding | int         | NO   |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```



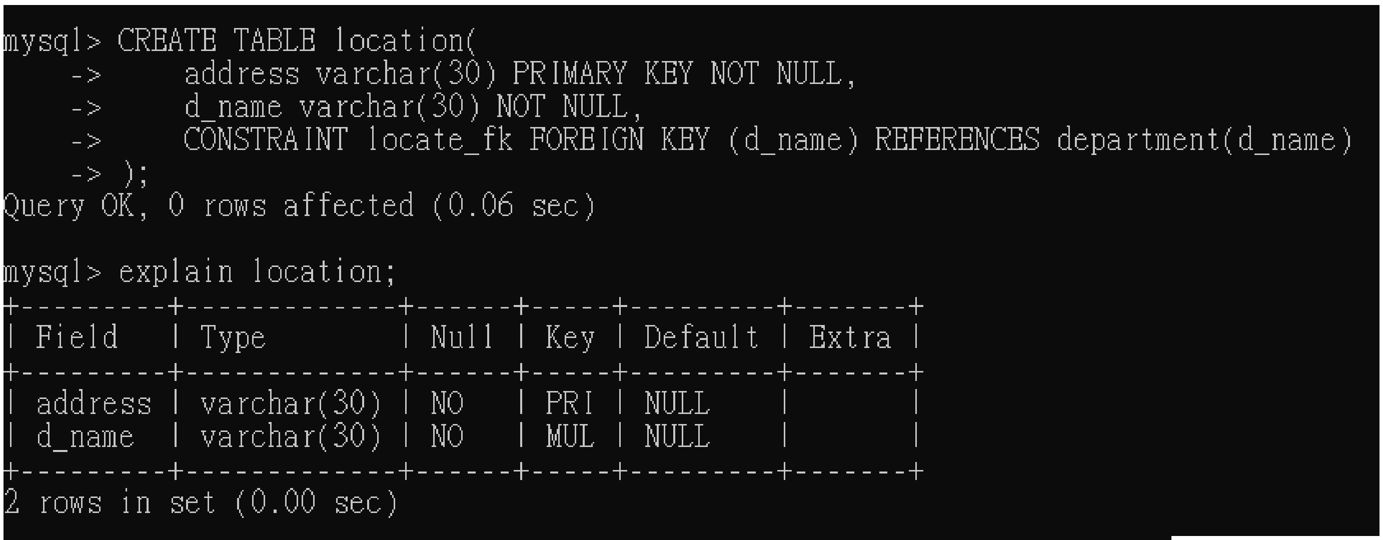
```
mysql> CREATE TABLE student (
    ->     NetID int PRIMARY KEY NOT NULL,
    ->     status varchar(30) NOT NULL,
    ->     d_name varchar(30) NOT NULL,
    ->     CONSTRAINT user_student_fk FOREIGN KEY (NetID) REFERENCES user(NetID),
    ->     CONSTRAINT depart_student_fk FOREIGN KEY (d_name)
    ->     REFERENCES department(d_name)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> explain student;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| NetID  | int         | NO   | PRI | NULL    |       |
| status | varchar(30) | NO   |     | NULL    |       |
| d_name | varchar(30) | NO   | MUL | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```



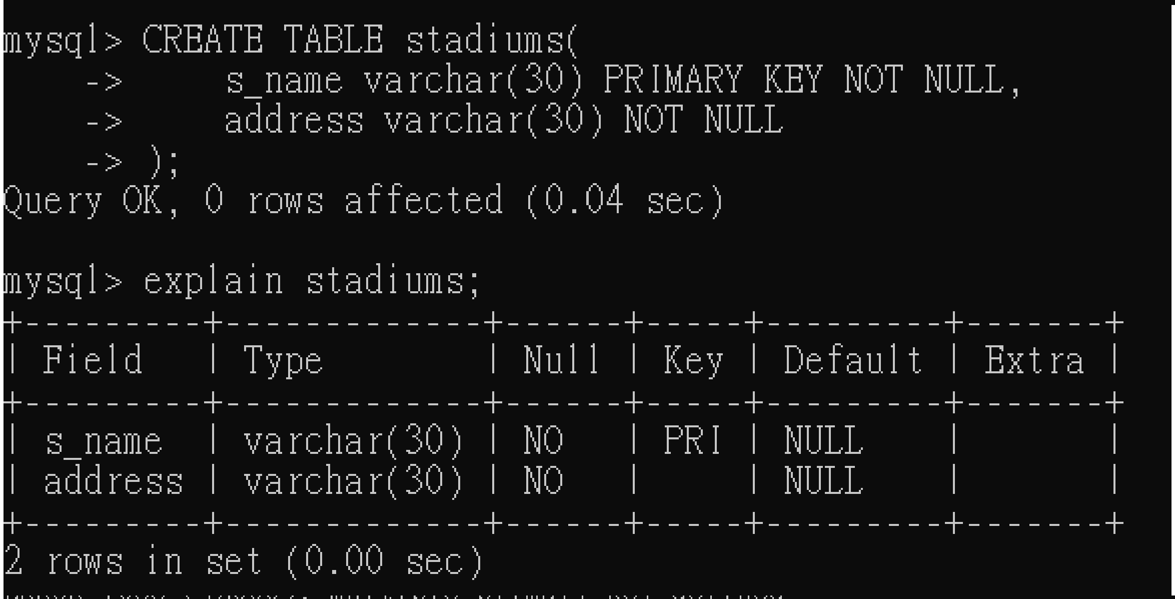
```
    CONSTRAINT user_professor_fk FOREIGN KEY (NetID)' at line 5
mysql> CREATE TABLE professor (
    ->     NetID int PRIMARY KEY NOT NULL,
    ->     p_rank varchar(30) NOT NULL,
    ->     CONSTRAINT user_professor_fk FOREIGN KEY (NetID) REFERENCES user(NetID)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> explain professor;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| NetID  | int         | NO   | PRI | NULL    |       |
| p_rank | varchar(30) | NO   |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
2 rows in set (0.02 sec)

mysql> CREATE TABLE staff (
    ->     NetID int PRIMARY KEY NOT NULL,
    ->     title varchar(30) NOT NULL,
    ->     CONSTRAINT user_staff_fk FOREIGN KEY (NetID) REFERENCES user(NetID)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> explain staff;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| NetID | int         | NO   | PRI | NULL    |       |
| title | varchar(30) | NO   |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set (0.02 sec)
```



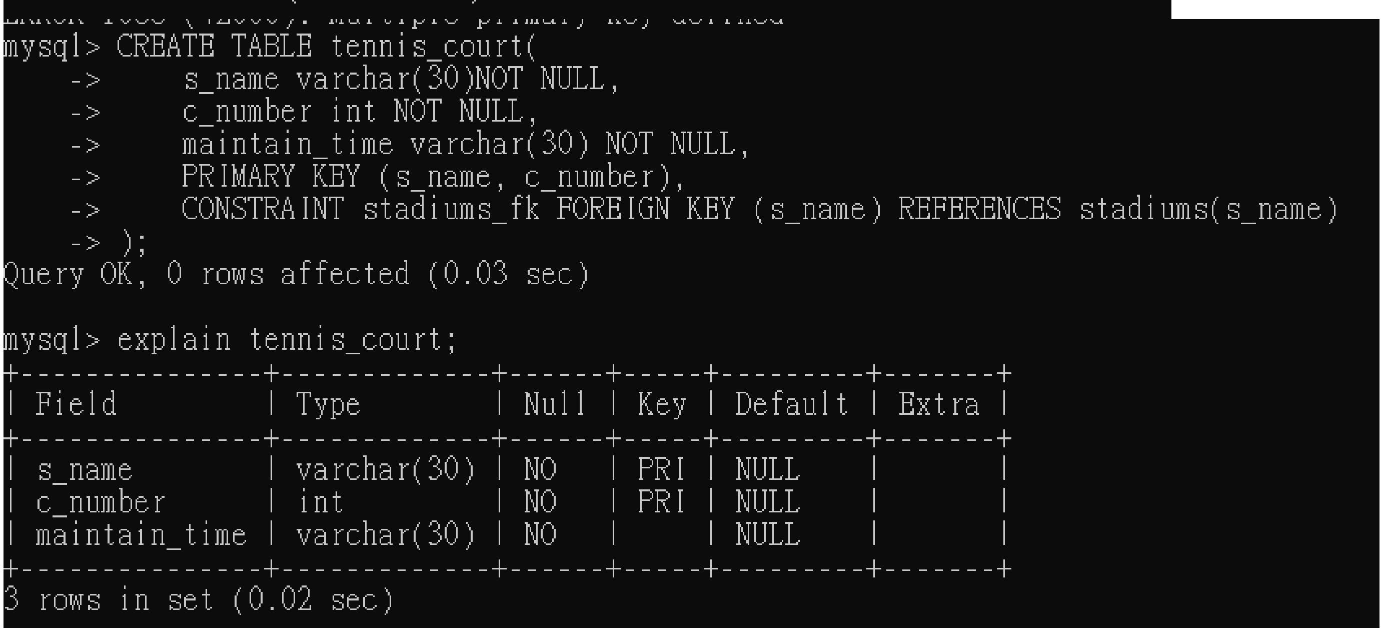
```
    FOREIGN KEY (d_name) REFERE' at line 5
mysql> CREATE TABLE work_in(
    ->     NetID int NOT NULL,
    ->     d_name varchar(30)NOT NULL,
    ->     PRIMARY KEY (NetId, d_name),
    ->     FOREIGN KEY (NetID) REFERENCES professor(NetID),
    ->     FOREIGN KEY (d_name) REFERENCES department(d_name)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> explain work_in;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| NetID  | int         | NO   | PRI | NULL    |       |
| d_name | varchar(30) | NO   | PRI | NULL    |       |
+--------+-------------+------+-----+---------+-------+
2 rows in set (0.02 sec)
```



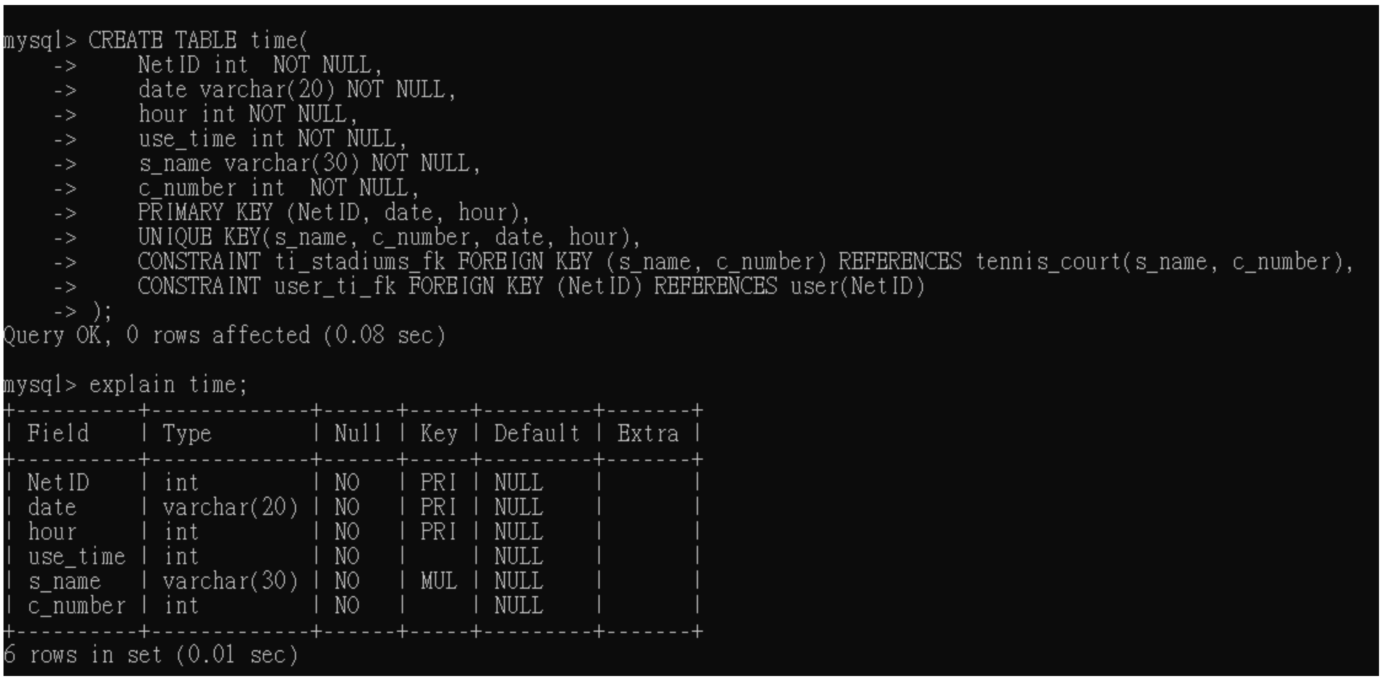
```
mysql> CREATE TABLE location(
    ->     address varchar(30) PRIMARY KEY NOT NULL,
    ->     d_name varchar(30) NOT NULL,
    ->     CONSTRAINT locate_fk FOREIGN KEY (d_name) REFERENCES department(d_name)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> explain location;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| address | varchar(30) | NO   | PRI | NULL    |       |
| d_name  | varchar(30) | NO   | MUL | NULL    |       |
+---------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```



```
mysql> CREATE TABLE stadiums(
    ->     s_name varchar(30) PRIMARY KEY NOT NULL,
    ->     address varchar(30) NOT NULL
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> explain stadiums;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| s_name  | varchar(30) | NO   | PRI | NULL    |       |
| address | varchar(30) | NO   |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```



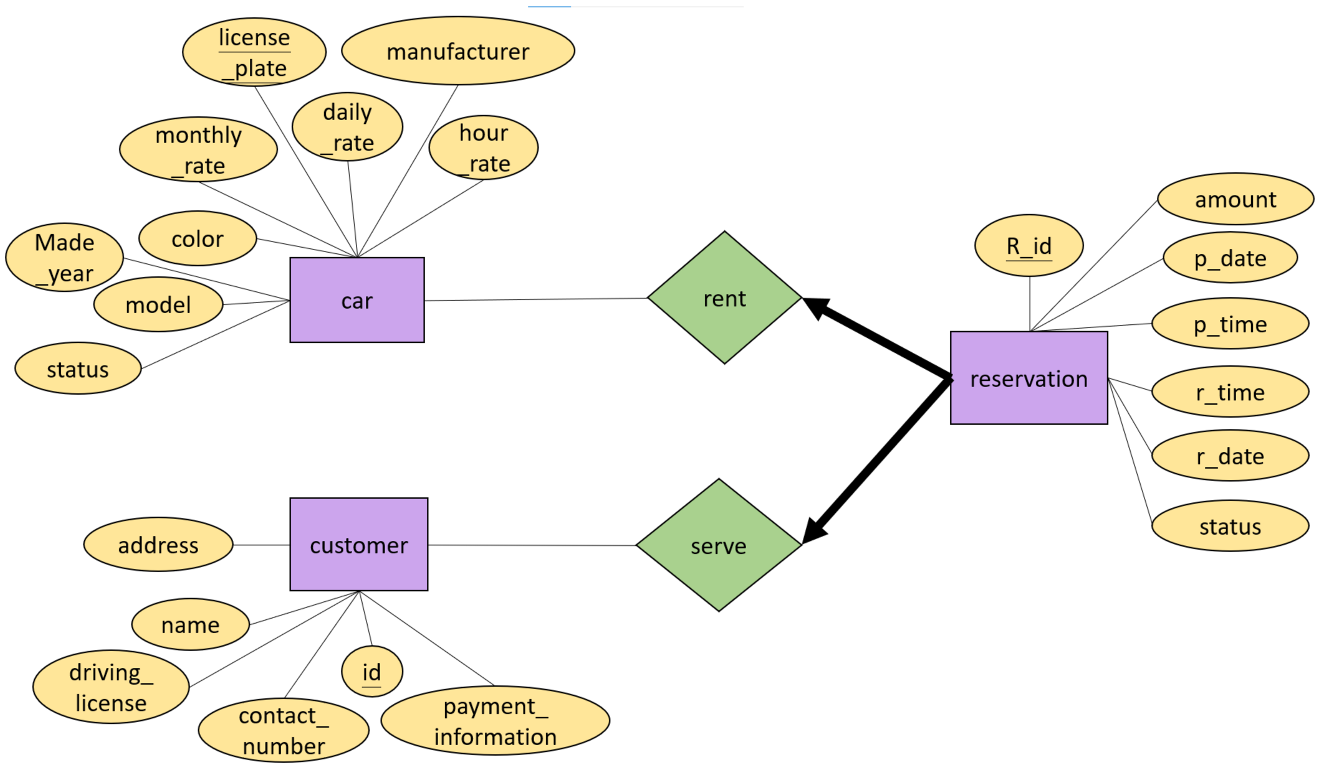
```
ERROR 1068 (42000): Multiple primary key defined
mysql> CREATE TABLE tennis_court(
    ->     s_name varchar(30)NOT NULL,
    ->     c_number int NOT NULL,
    ->     maintain_time varchar(30) NOT NULL,
    ->     PRIMARY KEY (s_name, c_number),
    ->     CONSTRAINT stadiums_fk FOREIGN KEY (s_name) REFERENCES stadiums(s_name)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> explain tennis_court;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| s_name        | varchar(30) | NO   | PRI | NULL    |       |
| c_number      | int         | NO   | PRI | NULL    |       |
| maintain_time | varchar(30) | NO   |     | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
3 rows in set (0.02 sec)
```



```
mysql> CREATE TABLE time(
    ->     NetID int  NOT NULL,
    ->     date varchar(20) NOT NULL,
    ->     hour int NOT NULL,
    ->     use_time int NOT NULL,
    ->     s_name varchar(30) NOT NULL,
    ->     c_number int  NOT NULL,
    ->     PRIMARY KEY (NetID, date, hour),
    ->     UNIQUE KEY(s_name, c_number, date, hour),
    ->     CONSTRAINT ti_stadiums_fk FOREIGN KEY (s_name, c_number) REFERENCES tennis_court(s_name, c_number),
    ->     CONSTRAINT user_ti_fk FOREIGN KEY (NetID) REFERENCES user(NetID)
    -> );
Query OK, 0 rows affected (0.08 sec)

mysql> explain time;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| NetID    | int         | NO   | PRI | NULL    |       |
| date     | varchar(20) | NO   | PRI | NULL    |       |
| hour     | int         | NO   | PRI | NULL    |       |
| use_time | int         | NO   |     | NULL    |       |
| s_name   | varchar(30) | NO   | MUL | NULL    |       |
| c_number | int         | NO   |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
6 rows in set (0.01 sec)
```

part9

ER-diagram



假設

1. 一輛車只記錄一種顏色
2. 一輛車只有一個製造商
3. car與reservation的狀態都只記最新的
4. 每一個顧客只記一份payment_information, address, contact number

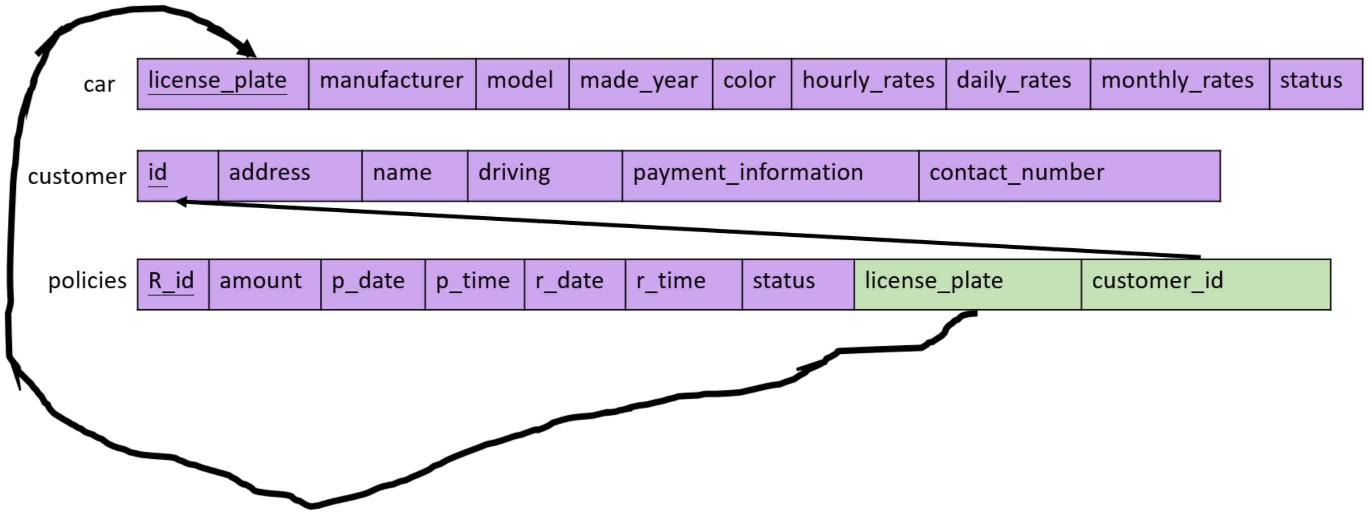
entity

car : 一輛車要記錄license plate, manufacturer, model, made year, color, hourly, daily,monthly rates和status。由於對於每輛車來說這些值都是唯一的，所以car為entity有以上9個attritube，且題目有說到我們可以透過license plate辨認車子，所以我們可以把license_plate當作car的key
customer : 每位顧客要記錄 customer id, name, address, contact number, driving license number和payment information。由於對於每個顧客來說這些值都是唯一的，所以customer為entity有以上6個attritube，且題目有說到可以用id辨識客人，顧id是customer的key。
reservation :每筆預約訂單要記錄R_id(unique
identification number), p_date(pick up date), p_time(pick up time) , r_date(return date), r_time(return time), amount, status，由於每個值都只有一個，所以都可以變成reservation的attritube(共7個attritube)，其中以R_id當key，因為他可以保證唯一。

relation

rent : 記錄車與預約的關係。因為每個預約都必須恰好有一台車，所以reservation與rent間有total participation以及key constraint。因為可能有車沒被預約過，或者被不同訂單預約，所以既沒有total participation，也沒有key constraint。
serve : 記錄顧客與預約的關係。因為每個預約都必須恰好有一為顧客，所以reservation與serve間有total participation以及key constraint。因為可能有人沒預約過，或者預約多次，所以既沒有total participation，也沒有key constraint。

relational schema



DDL

car

```
CREATE TABLE car(
    license_plate varchar(30) PRIMARY KEY NOT NULL,
    manufacturer  varchar(30) NOT NULL,
    model varchar(30) NOT NULL,
    made_year int NOT NULL,
    color varchar(30) NOT NULL,
    hourly_rates int NOT NULL,
    daily_rates int NOT NULL,
    monthly_rates int NOT NULL,
    status varchar(30) NOT NULL
);
```

customer

```
CREATE TABLE customer(
    id int PRIMARY KEY NOT NULL,
    address varchar(50),
    name varchar(30) NOT NULL,
    driving_license varchar(30) NOT NULL,
    payment_information varchar(30) NOT NULL,
    contact_number varchar(20) NOT NULL
);
```

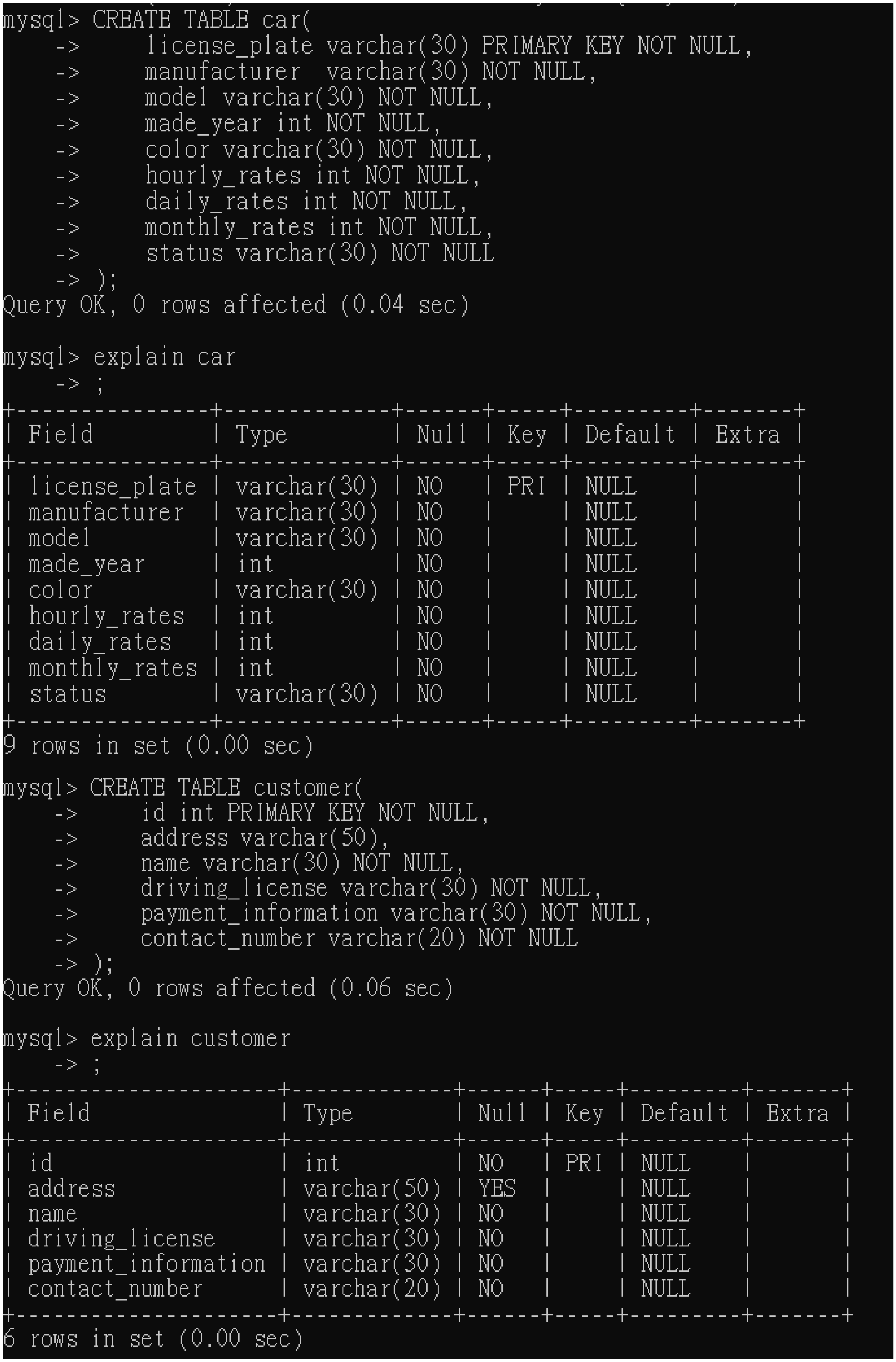
reservation

```
CREATE TABLE reservation(
    R_id int PRIMARY KEY NOT NULL,
    amount int NOT NULL,
    p_date varchar(30) NOT NULL,
    p_time varchar(30) NOT NULL,
    r_date varchar(30) NOT NULL,
    r_time varchar(30) NOT NULL,
    status varchar(30) NOT NULL,
    license_plate varchar(30) NOT NULL,
    customer_id int NOT NULL,
    CONSTRAINT rent_fk FOREIGN KEY (license_plate) REFERENCES car(license_plate),
    CONSTRAINT serve_fk FOREIGN KEY (customer_id) REFERENCES customer(id)
);
```

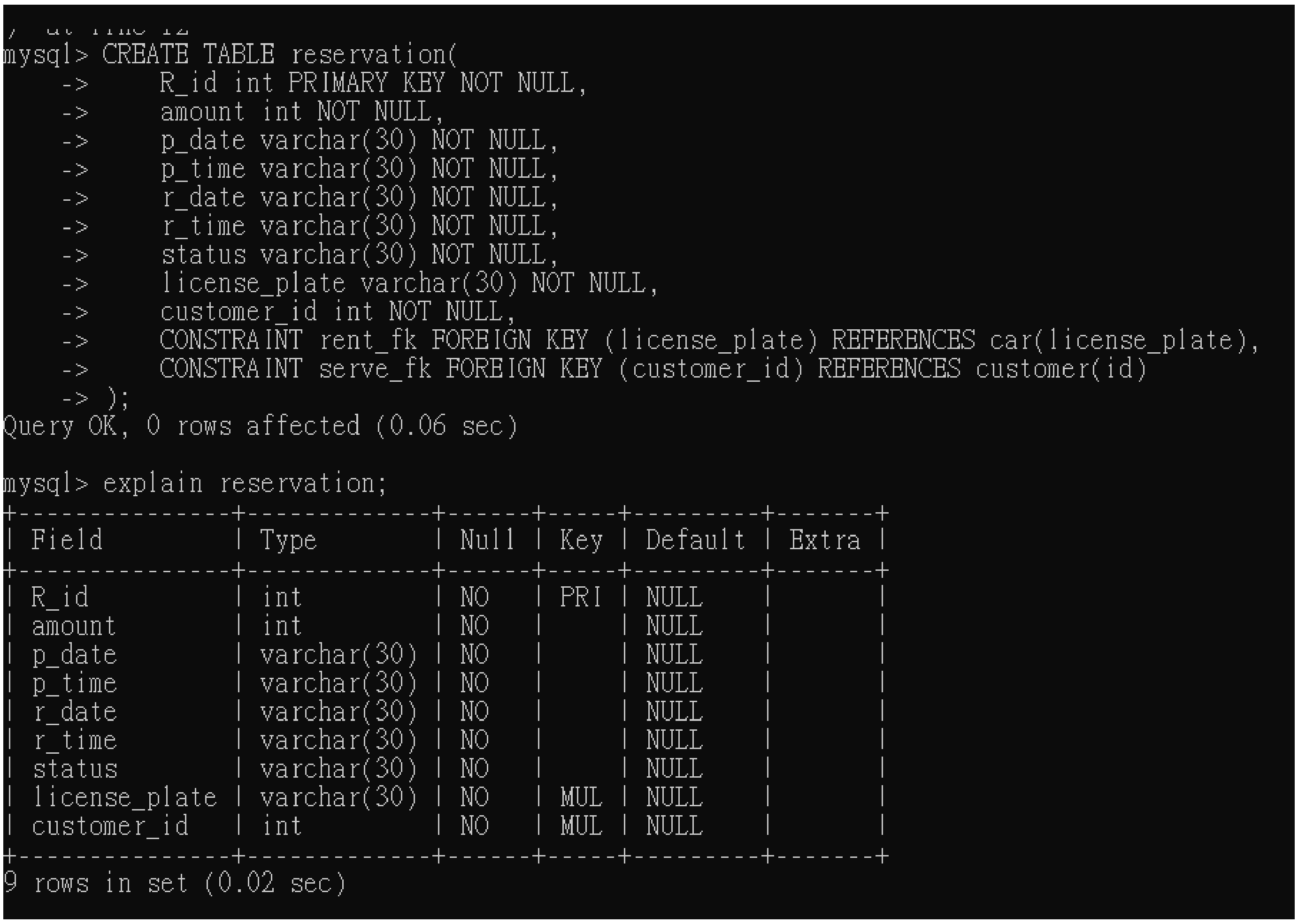
car : car是strong entity所以直接將所有attritude變成column，並將key設為license_plate(ER-diagram上car的key)
customer : customer是strong entity所以直接將所有attritude變成column，並將key設為id(ER-diagram上customer的key)
reservation : reservation是strong entity，所以直接將attritube變成column，並將R_id設為key(ER-diagram上reservation的key)。
rent : 由於他是一(car)對多(reservation)的relation，所以直接把car的key放到reservation的

table中，並設置foreign key，避免reservation訂到不在資料庫中的車，由於有total participation，所以reservation中的license_plate為not null

serve：由於他是一(customer)對多(reservation)的relation，所以直接把customer的key放到reservation的table中(customer_id)，並設置foreign key，避免reservation與不在資料庫內的客人訂定預約，由於有total participation，所以reservation中的customer_id為not null



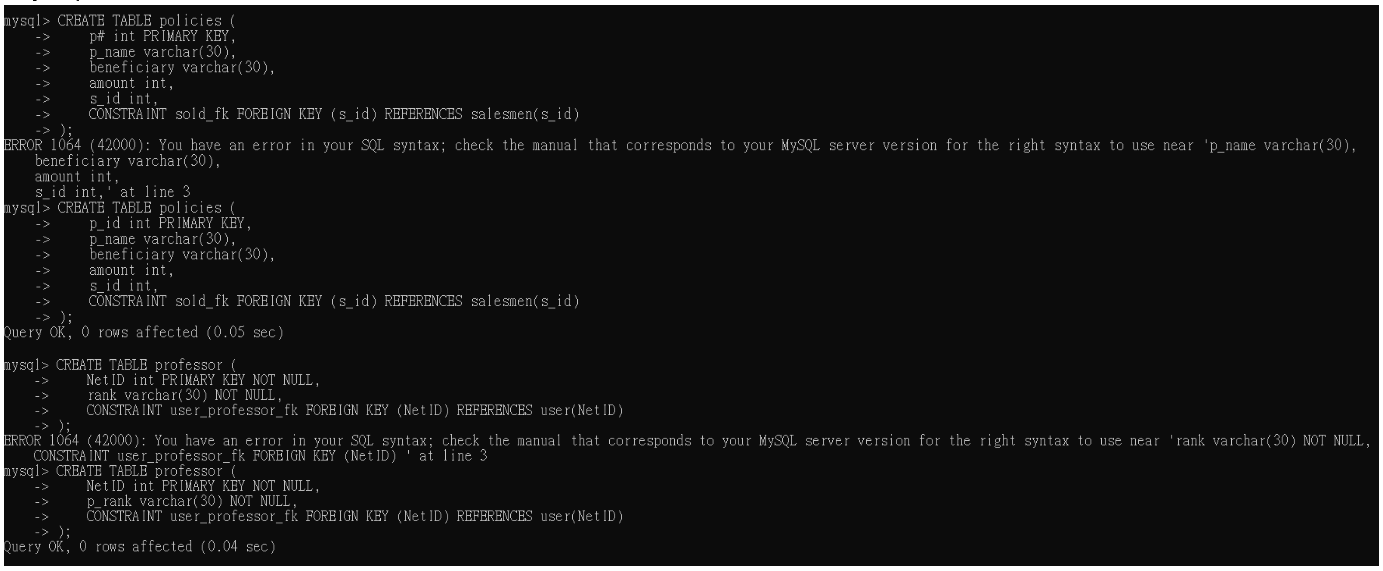
```
mysql> CREATE TABLE car(
    ->     license_plate varchar(30) PRIMARY KEY NOT NULL,
    ->     manufacturer  varchar(30) NOT NULL,
    ->     model varchar(30) NOT NULL,
    ->     made_year int NOT NULL,
    ->     color varchar(30) NOT NULL,
    ->     hourly_rates int NOT NULL,
    ->     daily_rates int NOT NULL,
    ->     monthly_rates int NOT NULL,
    ->     status varchar(30) NOT NULL
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> explain car
    -> ;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| license_plate | varchar(30) | NO   | PRI | NULL    |       |
| manufacturer  | varchar(30) | NO   |     | NULL    |       |
| model         | varchar(30) | NO   |     | NULL    |       |
| made_year     | int         | NO   |     | NULL    |       |
| color         | varchar(30) | NO   |     | NULL    |       |
| hourly_rates  | int         | NO   |     | NULL    |       |
| daily_rates   | int         | NO   |     | NULL    |       |
| monthly_rates | int         | NO   |     | NULL    |       |
| status        | varchar(30) | NO   |     | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
9 rows in set (0.00 sec)

mysql> CREATE TABLE customer(
    ->     id int PRIMARY KEY NOT NULL,
    ->     address varchar(50),
    ->     name varchar(30) NOT NULL,
    ->     driving_license varchar(30) NOT NULL,
    ->     payment_information varchar(30) NOT NULL,
    ->     contact_number varchar(20) NOT NULL
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> explain customer
    -> ;
+---------------------+-------------+------+-----+---------+-------+
| Field               | Type        | Null | Key | Default | Extra |
+---------------------+-------------+------+-----+---------+-------+
| id                  | int         | NO   | PRI | NULL    |       |
| address             | varchar(50) | YES  |     | NULL    |       |
| name                | varchar(30) | NO   |     | NULL    |       |
| driving_license     | varchar(30) | NO   |     | NULL    |       |
| payment_information | varchar(30) | NO   |     | NULL    |       |
| contact_number      | varchar(20) | NO   |     | NULL    |       |
+---------------------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```



```
mysql> CREATE TABLE reservation(
    ->     R_id int PRIMARY KEY NOT NULL,
    ->     amount int NOT NULL,
    ->     p_date varchar(30) NOT NULL,
    ->     p_time varchar(30) NOT NULL,
    ->     r_date varchar(30) NOT NULL,
    ->     r_time varchar(30) NOT NULL,
    ->     status varchar(30) NOT NULL,
    ->     license_plate varchar(30) NOT NULL,
    ->     customer_id int NOT NULL,
    ->     CONSTRAINT rent_fk FOREIGN KEY (license_plate) REFERENCES car(license_plate),
    ->     CONSTRAINT serve_fk FOREIGN KEY (customer_id) REFERENCES customer(id)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> explain reservation;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| R_id          | int         | NO   | PRI | NULL    |       |
| amount        | int         | NO   |     | NULL    |       |
| p_date        | varchar(30) | NO   |     | NULL    |       |
| p_time        | varchar(30) | NO   |     | NULL    |       |
| r_date        | varchar(30) | NO   |     | NULL    |       |
| r_time        | varchar(30) | NO   |     | NULL    |       |
| status        | varchar(30) | NO   |     | NULL    |       |
| license_plate | varchar(30) | NO   | MUL | NULL    |       |
| customer_id   | int         | NO   | MUL | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
9 rows in set (0.02 sec)
```

問題排解

mysql無法以 `#`, `rank` 命名

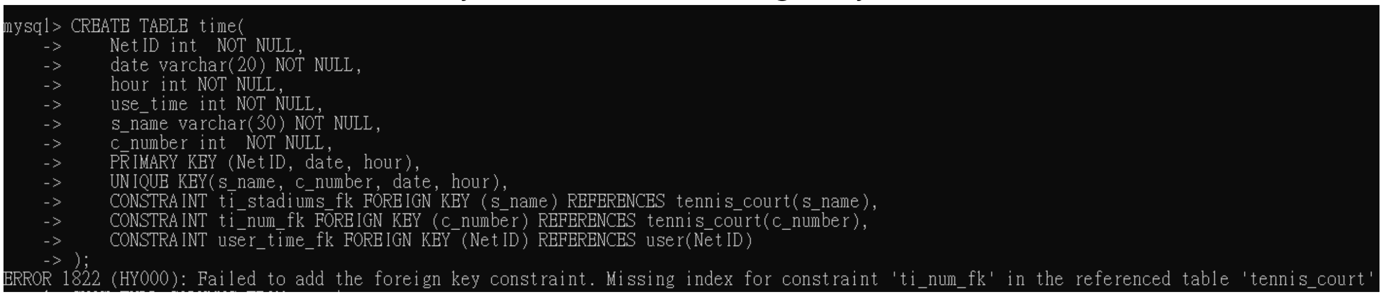


```
mysql> CREATE TABLE policies (
    ->     p# int PRIMARY KEY,
    ->     p_name varchar(30),
    ->     beneficiary varchar(30),
    ->     amount int,
    ->     s_id int,
    ->     CONSTRAINT sold_fk FOREIGN KEY (s_id) REFERENCES salesmen(s_id)
    -> );
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'p_name varchar(30),
    beneficiary varchar(30),
    amount int,
    s_id int,' at line 3
mysql> CREATE TABLE policies (
    ->     p_id int PRIMARY KEY,
    ->     p_name varchar(30),
    ->     beneficiary varchar(30),
    ->     amount int,
    ->     s_id int,
    ->     CONSTRAINT sold_fk FOREIGN KEY (s_id) REFERENCES salesmen(s_id)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE professor (
    ->     NetID int PRIMARY KEY NOT NULL,
    ->     rank varchar(30) NOT NULL,
    ->     CONSTRAINT user_professor_fk FOREIGN KEY (NetID) REFERENCES user(NetID)
    -> );
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'rank varchar(30) NOT NULL,
    CONSTRAINT user_professor_fk FOREIGN KEY (NetID) ' at line 3
mysql> CREATE TABLE professor (
    ->     NetID int PRIMARY KEY NOT NULL,
    ->     p_rank varchar(30) NOT NULL,
    ->     CONSTRAINT user_professor_fk FOREIGN KEY (NetID) REFERENCES user(NetID)
    -> );
Query OK, 0 rows affected (0.04 sec)
```

rank 為mysql 語法之一

解法:換個名字

tennis_court的c_number不是key，無法用它來做foreign key



```
mysql> CREATE TABLE time(
    ->     NetID int  NOT NULL,
    ->     date varchar(20) NOT NULL,
    ->     hour int NOT NULL,
    ->     use_time int NOT NULL,
    ->     s_name varchar(30) NOT NULL,
    ->     c_number int  NOT NULL,
    ->     PRIMARY KEY (NetID, date, hour),
    ->     UNIQUE KEY(s_name, c_number, date, hour),
    ->     CONSTRAINT ti_stadiums_fk FOREIGN KEY (s_name) REFERENCES tennis_court(s_name),
    ->     CONSTRAINT ti_num_fk FOREIGN KEY (c_number) REFERENCES tennis_court(c_number),
    ->     CONSTRAINT user_time_fk FOREIGN KEY (NetID) REFERENCES user(NetID)
    -> );
ERROR 1822 (HY000): Failed to add the foreign key constraint. Missing index for constraint 'ti_num_fk' in the referenced table 'tennis_court'
```

因為tennis 的key是s_name+c_number，所以c_number不能唯一指定一個tennis_court，所以foreign constraint會失敗，找不到唯一一個球場。

解決:把s_name和c_number綁再一起去做foreign key。

reference

https://www.itread01.com/content/1546832286.html (https://www.itread01.com/content/1546832286.html)

https://discuss.codecademy.com/t/error-code-1822-failed-to-add-the-foreign-key-constraint-missing-index-for-constraint-employees-ibfk-1-in-the-referenced-table-departments-0-000-sec/559058 (https://discuss.codecademy.com/t/error-code-1822-failed-to-add-the-foreign-key-constraint-missing-index-for-constraint-employees-ibfk-1-in-the-referenced-table-departments-0-000-sec/559058)

https://stackoverflow.com/questions/26329775/error-code-1822-failed-to-add-the-foreign-key-constaint-missing-index-for-con (https://stackoverflow.com/questions/26329775/error-code-1822-failed-to-add-the-foreign-key-constaint-missing-index-for-con)