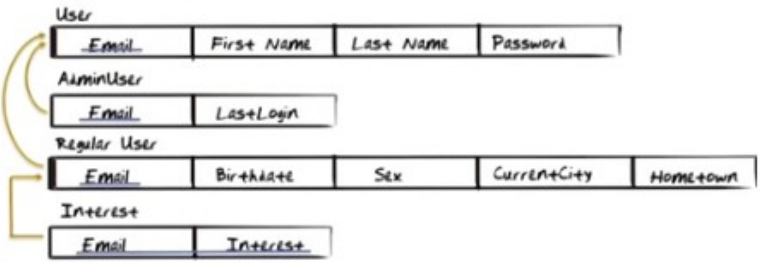
Java p1204: char(5) specifis that *courseID* consistsof 5 characters. Varchar(50) specifies that *title* is a variant-length string with a maximum of 50 characters.

注意看本 note 時, 要打開 6. Methodology III DESIGN. 好看以下每個 relation 表對應的 EER 圖.

本 note 跟 SQL 無關, 以下的 SQL statements 當偽碼看就可以了. 之後會專門講 SQL.

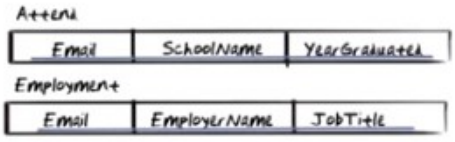
SQL: Create Table Statements




```
CREATE TABLE `User`(
        Email varchar(50) NOT NULL,
        FirstName varchar(50) NOT NULL,
        LastName varchar(50) NOT NULL,
        Password varchar(50) NOT NULL,
        PRIMARY KEY (Email));

CREATE TABLE RegularUser(
        Email varchar(50) NOT NULL,
        Sex char(1) NULL,
        Birthdate datetime NULL,
        CurrentCity varchar(50) NULL,
        Hometown varchar(50) NULL,
        PRIMARY KEY (Email),
        FOREIGN KEY (Email)
            REFERENCES `User` (Email));

CREATE TABLE AdminUser(
        Email varchar(50) NOT NULL,
        LastLogin datetime NULL,
        PRIMARY KEY (Email),
        FOREIGN KEY (Email)
            REFERENCES `User` (Email));

CREATE TABLE Interests(
        Email varchar(50) NOT NULL,
        Interest varchar(50) NOT NULL,
        PRIMARY KEY (Email,    Interest),
        FOREIGN KEY (Email)
        REFERENCES RegularUser (Email));
```

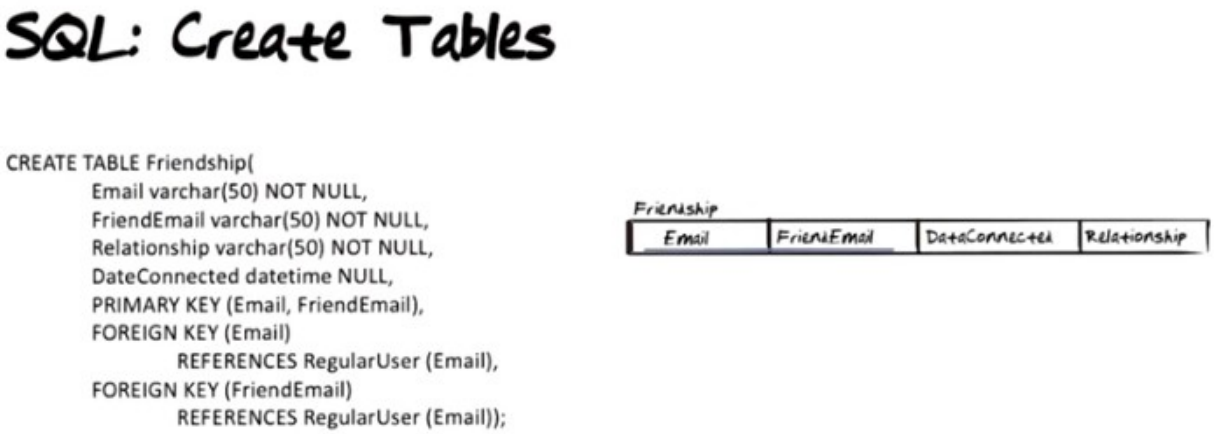
上圖中 NOT NULL 意思是 not allowed to be null. NULL 意思是 initially set to NULL. 注意在 mysql 中, 都是小寫(如 create table 等).

SQL: Create Tables

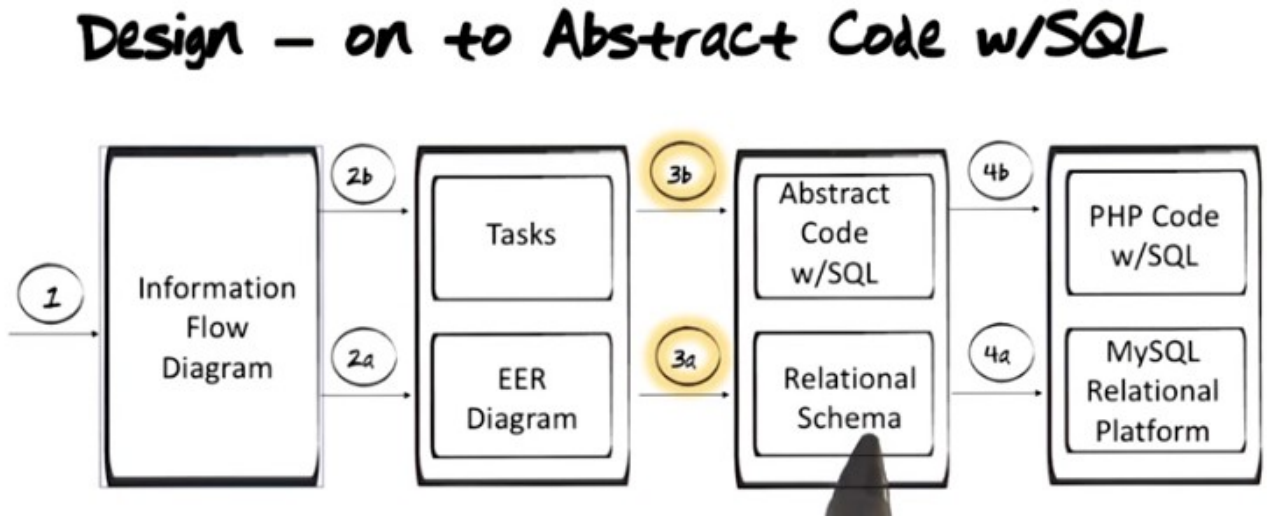



```
CREATE TABLE Attend(
        Email varchar(50) NOT NULL,
        SchoolName varchar(50) NOT NULL,
        YearGraduated int NULL,
        UNIQUE (Email, SchoolName, YearGraduated),
        FOREIGN KEY (Email) REFERENCES RegularUser (Email))
        FOREIGN KEY (SchoolName) REFERENCES School (SchoolName));

CREATE TABLE Employment(
        Email varchar(50) NOT NULL,
        EmployerName varchar(50) NOT NULL,
        JobTitle varchar(50) NOT NULL,
        UNIQUE (Email, EmployerName, JobTitle),
        FOREIGN KEY (EmployerName) REFERENCES Employer (EmployerName),
        FOREIGN KEY (Email) REFERENCES RegularUser (Email));
```

上圖中的 Attend 和 Employment 都是 EER 中的 relationship. 注意以上多了個 UNIQUE 語句, 但沒有 PRIMARY KEY. 這是因為: the customer requirements that we are working from allows YearGraduated to be left blank. Because no portion of a PRIMARY KEY is allowed to be NULL, it is not possible to use primary key in the table definition. So I'm using UNIQUE instead. UNIQUE 之意思見 textbook p151.



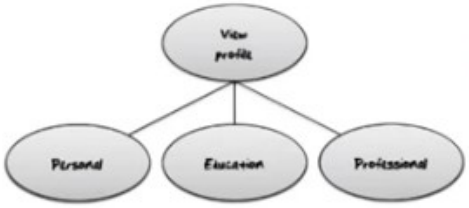
The above figure: Relationship is NOT NULL because the value is required, not because it's part of the primary key.

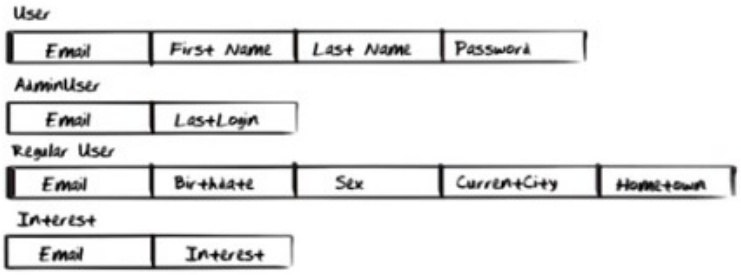


上圖的意思是, 我們現在已經完成了 3a, 下面我們來弄 3b.

SQL: View Profile (Part 1)

```
//assume $Email of current user is managed by
application

SELECT FirstName, LastName, Birthdate, Sex, CurrentCity,
Hometown
FROM User INNER JOIN RegularUser ON
User.Email = RegularUser.Email
WHERE Email=$Email;

SELECT Interest
FROM Interests
WHERE Email=$Email;
```

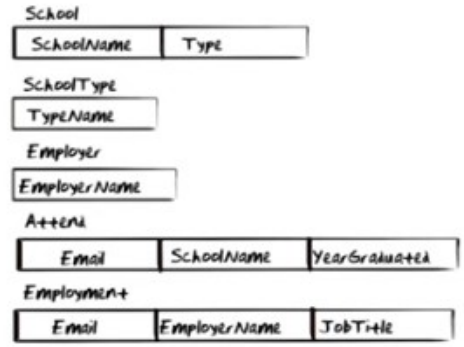


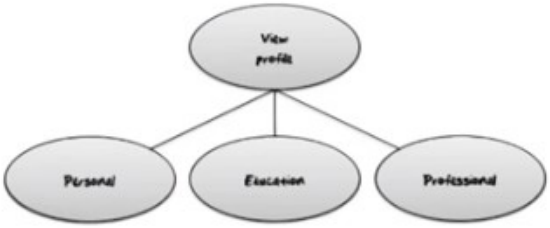


上圖是弄 sub task Personal. 其中的
FROM User INNER JOIN RegularUser ON
User.Email = RegularUser.Email
的意思是: 在 User 並 RegularUSer 中, 按 User.Email = RegularUser.Email 去找, 難得說清, 但很容易想明白.

SQL: View Profile (Part 2)

```
SELECT School.SchoolName, School.Type,
Attend.YearGraduated
FROM Attend INNER JOIN School ON
Attend.SchoolName=School.SchoolName
WHERE Email=$Email
ORDER BY Attend.YearGraduated DESC;

SELECT EmployerName, JobTitle
FROM Employment
WHERE Email=$Email;
```

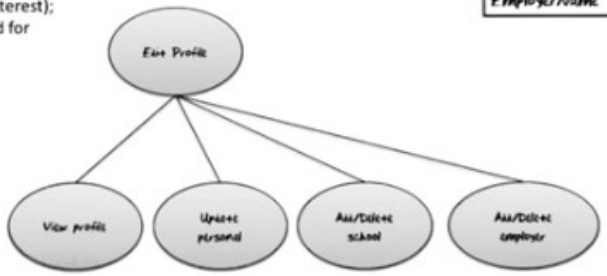




上圖是弄 sub task Education 和 Professional. 注意上圖中, 表中元素的下劃線都省掉了. Attend 中, Email, SchoolName, YearGradauted 本來都有下劃線, School 中的 SchoolName 也有下劃線. 其餘的類似.

SQL: Edit Profile (Part 1)

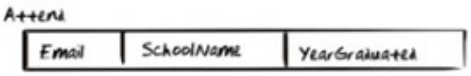
```
//populate dropdowns
SELECT SchoolName, Type FROM School;
SELECT EmployerName FROM Employer;

//assume $Email of current user is managed by application
//read $Sex, $Birthdate, $CurrentCity, $Hometown
UPDATE RegularUser
SET Sex=$Sex, Birthdate=$Birthdate, CurrentCity=$CurrentCity,
Hometown=$Hometown
WHERE Email=$Email;

for each $Interest
INSERT INTO Interest (Email, Interest) VALUES ($Email,
$Interest);
end for
```



上圖中的//read $Sex, $Birthdate, $CurrentCity, $Hometown 即從那個 GUI application 的文字框(the input text field)中讀入 Sex, Birthdate, CurrentCity, Hometown 的值.

SQL: Edit Profile (Part 2)

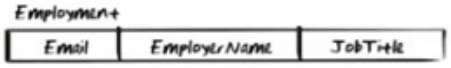
```
//if user clicks "Delete this School"
DELETE FROM Attend
WHERE Email=$Email AND SchoolName=$SchoolName
        AND YearGraduated=$YearGraduated

//add new schools
for each $SchoolName, $YearGraduated
INSERT INTO Attend (Email, SchoolName, YearGraduated)
        VALUES ($Email, $SchoolName, $YearGraduated)
 end for

//update existing schools
//assume application manages $OldSchoolName, $OldYearGraduated
for each $SchoolName, $YearGraduated
UPDATE Attend
SET SchoolName=$SchoolName, YearGraduated=$YearGraduated
WHERE Email=$Email AND SchoolName=$OldSchoolName
        AND YearGraduated=$OldYearGraduated
 end for
```

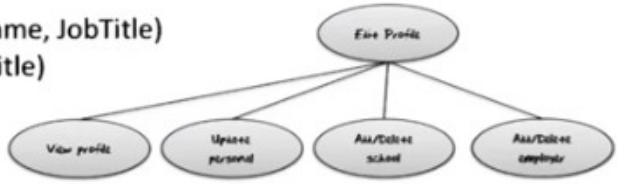


上圖中的 for each $SchoolName, $YearGraduated 也是從 GUI application 中讀入的. 後同.

SQL: Edit Profile (Part 3)

```
//if user clicks "Delete this Job"
DELETE FROM Employment WHERE Email=$Email AND
EmployerName=$EmployerName AND JobTitle=$JobTitle

//add new jobs
for each $EmployerName, $JobTitle
INSERT INTO Employment (Email, EmployerName, JobTitle)
      VALUES ($Email, $EmployerName, $JobTitle)
 end for

//update existing jobs
//assume application manages $OldEmployerName, $OldJobTitle
for each $EmployerName, $JobTitle
UPDATE Employment
SET EmployerName=$EmployerName, JobTitle=$JobTitle
WHERE Email=$Email AND EmployerName=$OldEmployerName
            AND JobTitle=$OldJobTitle
 end for
```

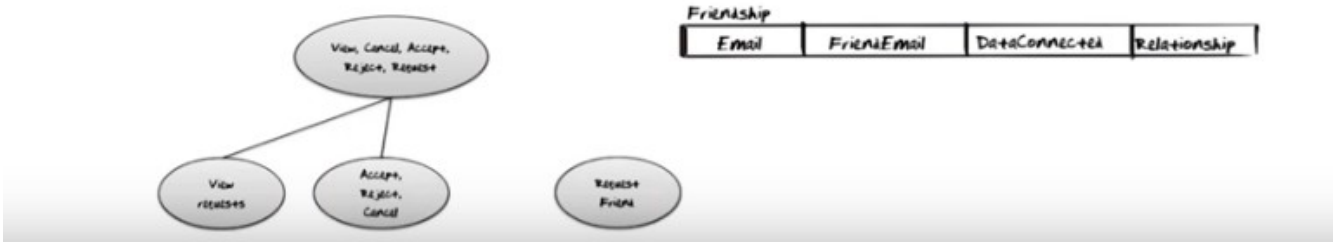




上圖中的 EmployerName=$EmployerName AND JobTitle=$JobTitle 也是從 GUI application 中讀. 後同.

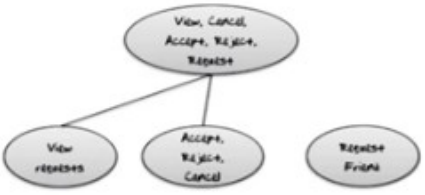
SQL: Request Friend

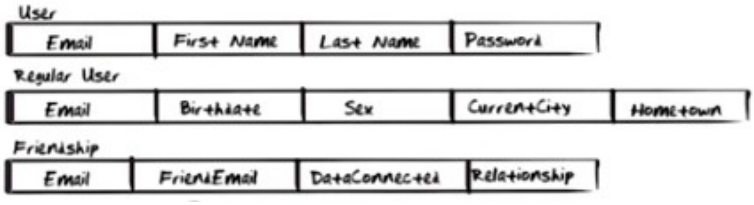
```
//application provides $FriendEmail (based on user
click)
//read $Relationship from user
INSERT INTO Friendship
(Email, FriendEmail, DateConnected, Relationship)
VALUES ($Email, $FriendEmail, NULL, $Relationship)
```



SQL: View Requests

```
//application provides $Email of current user
//show pending requests to you
SELECT R.Email, FirstName, LastName, Hometown, Relationship
FROM Friendship AS F INNER JOIN RegularUser AS R ON F.Email = R.Email INNER
JOIN User AS U ON U.Email = R.Email
WHERE F.FriendEmail=$Email AND DateConencted IS NULL

//show pending requests from you
SELECT R.Email, FirstName, LastName, Hometown, Relationship
FROM Friendship AS F INNER JOIN RegularUser AS R
ON F.FriendEmail = R.Email INNER JOIN User AS U
ON U.Email = R.Email WHERE F.Email=$Email AND DateConencted IS NULL
```





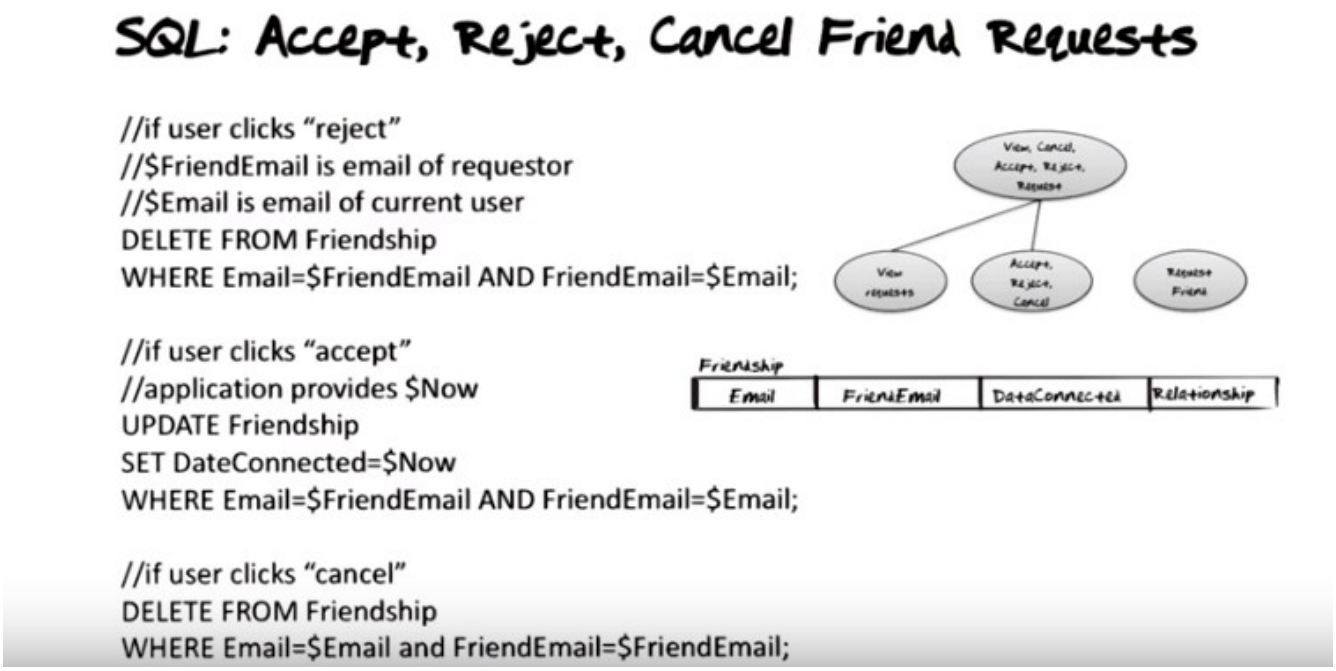
上圖中的
FROM Friendship AS F INNER JOIN RegularUser AS R ON F.Email = R.Email INNER JOIN User AS U ON U.Email = R.Email

意思即 join Friendship, RegularUser, User 這三個, 其中將 Friendship 取了個名字叫 F, 將 RegularUser 取了個名字叫 R, 將 User 取了個名字叫 U. 當然還要求它們 Email 相同.

注意這兩個例子中, 最終要顯示在 GUI 中的是 R.Email(見 SELECT R.Email 一句).

在前一個例子中(show pending request to you), GUI application 要顯示的是 '給你發好友申請的那個人' 的各種信息. 注意此時你的郵箱是 Friendship 表中的 FriendEmail, 而發 '給你發好友申請的那個人' 的郵箱才是 Friendship 表中的 Email. 這是 Friendship 自動弄好的, 沒得選. 所以在 FROM Friendship...一句中, 要求的是 F.Email = R.Email(R 即要顯示的那個人, 即 '給你發好友申請的那個人'), 而 WHERE 一句中, 要求的是 F.FriendEmail=$Email(即你的郵箱).

在第二個例子中(show pending request from you), GUI application 要顯示的是 '你想加為好友的那個人' 的各種信息. 注意此時你的郵箱是 Friendship 表中的 Email, 而發 '給你發好友申請的那個人' 的郵箱才是 Friendship 表中的 FriendEmail. 這是 Friendship 自動弄好的, 沒得選.所以有 F.FriendEmail=R.Email 和 F.Email=$Email.



上圖中的前兩個例子是處理 別人發給你的好友申請. 圖中 notation 用得有點 confusing. $FriendEmail 可改為更好理解的$HaoyouYouxiang, $Email 可改為$NideYouxiang.

第一個例子中, 在 DELETE 時, 當然要找到正確的刪, 注意此時還是別人申請加你為好友, 故 Friendship 表中的 FriendEmail 表示你的郵箱, Friendship 表中的 Email 表示 '給你發好友申請的那個人' 的郵箱. 這是 Friendship 自動弄好的, 沒得選. 所以在 WHERE 中要按這樣來檢查: WHERE Email=$HaoyouYouxiang AND FriendEmail=$NideYouxiang.

第二個例子跟第一個例子一樣理解.

第三個例子是 cancel 你發出的好友申請. 故 Friendship 表中的 Email 表示你的郵箱, Friendship 表中的 Friendship 表示 '給你發好友申請的那個人' 的郵箱. 這是 Friendship 自動弄好的, 沒得選. 所以在 WHERE

中要按這樣來檢查: WHERE Email=$ NideYouxiang AND FriendEmail=$ HaoyouYouxiang.