

# Final Project Report

0716002 李昕穎 0716008 呂沛儒 0716080 劉文心 0716203 黃子潔

## Data

- Tables & Columns

Table Name	Attribute Name	Description	Type	NULL
movie	<u>id</u>	title ID on IMDb	char(9)	NO
	title	name of the movie	varchar(110)	NO
	year	year of release	char(5)	
movie detail	<u>id</u>	title ID on IMDb	char(9)	NO
	duration	duration (in minutes)	smallint	
	country	movie country	varchar(50)	
director	<u>id</u>	title ID on IMDb	char(9)	NO
	<u>director</u>	director name	varchar(110)	NO
genre	<u>id</u>	title ID on IMDb	char(9)	NO
	<u>genre</u>	movie genre	varchar(15)	NO
all_gender	<u>id</u>	title ID on IMDb	char(9)	NO
	votes	number of votes received	int	NO
	rating	average rating	float	NO
	avg_0_18	average rating from users with age lower than 18	float	NO
	num_0_18	number of votes from users with age lower than 18	int	NO
	avg_18_30	average of votes from users with age greater or equal to 18 and lower than 30	float	NO
	num_18_30	number of votes from users with age greater or equal to 18	int	NO

		and lower than 30		
	avg_30_45	average of votes from users with age greater or equal to 30 and lower than 45	float	NO
	num_30_45	number of votes from users with age greater or equal to 30 and lower than 45	int	NO
	avg_45up	average rating from users with age greater or equal to 45	float	NO
	num_45up	number of votes from users with age greater or equal to 45	int	NO
male	<u>id</u>	title ID on IMDb	char(9)	NO
	votes	number of votes received	int	NO
	rating	average vote	float	NO
	avg_0_18	average rating from users with age lower than 18	float	NO
	num_0_18	number of votes from users with age lower than 18	int	NO
	avg_18_30	average of votes from users with age greater or equal to 18 and lower than 30	float	NO
	num_18_30	number of votes from users with age greater or equal to 18 and lower than 30	int	NO
	avg_30_45	average of votes from users with age greater or equal to 30 and lower than 45	float	NO
	num_30_45	number of votes from users with age greater or equal to 30 and lower than 45	int	NO
	avg_45up	average rating from users with age greater or equal to 45	float	NO
	num_45up	number of votes from users with age greater or equal to 45	int	NO
female	<u>id</u>	title ID on IMDb	char(9)	NO

	votes	number of votes received	int	NO
	rating	average vote	float	NO
	avg_0_18	average rating from users with age lower than 18	float	NO
	num_0_18	number of votes from users with age lower than 18	int	NO
	avg_18_30	average of votes from users with age greater or equal to 18 and lower than 30	float	NO
	num_18_30	number of votes from users with age greater or equal to 18 and lower than 30	int	NO
	avg_30_45	average of votes from users with age greater or equal to 30 and lower than 45	float	NO
	num_30_45	number of votes from users with age greater or equal to 30 and lower than 45	int	NO
	avg_45up	average rating from users with age greater or equal to 45	float	NO
	num_45up	number of votes from users with age greater or equal to 45	int	NO
golden_globe	<u>year</u>	year which the film screened at	char(5)	NO
	<u>nomination</u>	name of nomination	varchar(110)	NO
	<u>film</u>	film's title	varchar(110)	NO
	win	win the prize or not	varchar(10)	NO
oscar	<u>year</u>	year which the film screened at	char(5)	NO
	<u>nomination</u>	name of nomination	varchar(110)	NO
	<u>film</u>	film's title	varchar(110)	NO
	win	win the prize or not	varchar(10)	NO

- Data Source
  - IMDb movies extensive dataset
    - IMDB movies.csv
      - movie
      - movie\_detail
      - genre
      - director
    - IMDB ratings.csv
      - all\_gender
      - male
      - female
  - The Oscar Award, 1927 - 2020
    - the\_oscar\_award.csv
      - oscar
  - Golden Globe Awards, 1944 - 2020
    - golden\_globe\_awards.csv
      - golden\_globe
- Normalization
  - 3NF Definition

## Third Normal Form

- relation schema R is in 3NF if,
- for all  $X \rightarrow A$  (A is an attribute that holds over R
- $A \in X$  ( i.e.,  $X \rightarrow A$  is a trivial FD ), or
  - X is a superkey, or
  - A is part of some key for R
- The definition of 3NF is similar to that of BCNF, with the only difference being the third condition.  
     If R is in BCNF, obviously it is in 3NF.
  - A key for a relation is a minimal set of attributes that uniquely determines all other attributes.
    - A must be part of a key (any key, if there are several).

- Example

## Reserves

sid	bid	day	cardno
1	b1	April 5, 00	2323
2	b2	May 10, 01	1000
1	b3	January 5, 01	2323
3	b4	June 1, 00	4000

- Assume:  $\text{sid} \rightarrow \text{cardno}$  (a sailor uses a unique credit card to pay for reservations).
- Reserves is not in 3NF
  - $(\text{sid}, \text{bid}, \text{day})$  is the only key.
  - $\text{sid} \rightarrow \text{cardno}$  violates 3NF because
    - $\text{sid}$  is not a key and  $\text{cardno}$  is not part of a key
  - $(\text{sid}, \text{cardno})$  pairs are **redundantly stored**.

## Reserves

sid	bid	day	cardno
1	b1	April 5, 00	2323
2	b2	May 10, 01	1000
1	b3	January 5, 01	2323
3	b4	June 1, 00	4000

- Assume:  $\text{sid} \rightarrow \text{cardno}$ , and  $\text{cardno} \rightarrow \text{sid}$  (we know that credit cards also uniquely identify the owner).
- Reserves is in 3NF
  - $(\text{sid}, \text{bid}, \text{day})$  is a key for Reserves.
  - $(\text{cardno}, \text{bid}, \text{day})$  is also a key for Reserves.
  - $\text{cardno} \rightarrow \text{sid}$  does not violate 3NF, since  $\text{sid}$  is in a key
  - $\text{sid} \rightarrow \text{cardno}$  does not violate 3NF, since  $\text{cardno}$  is in a key

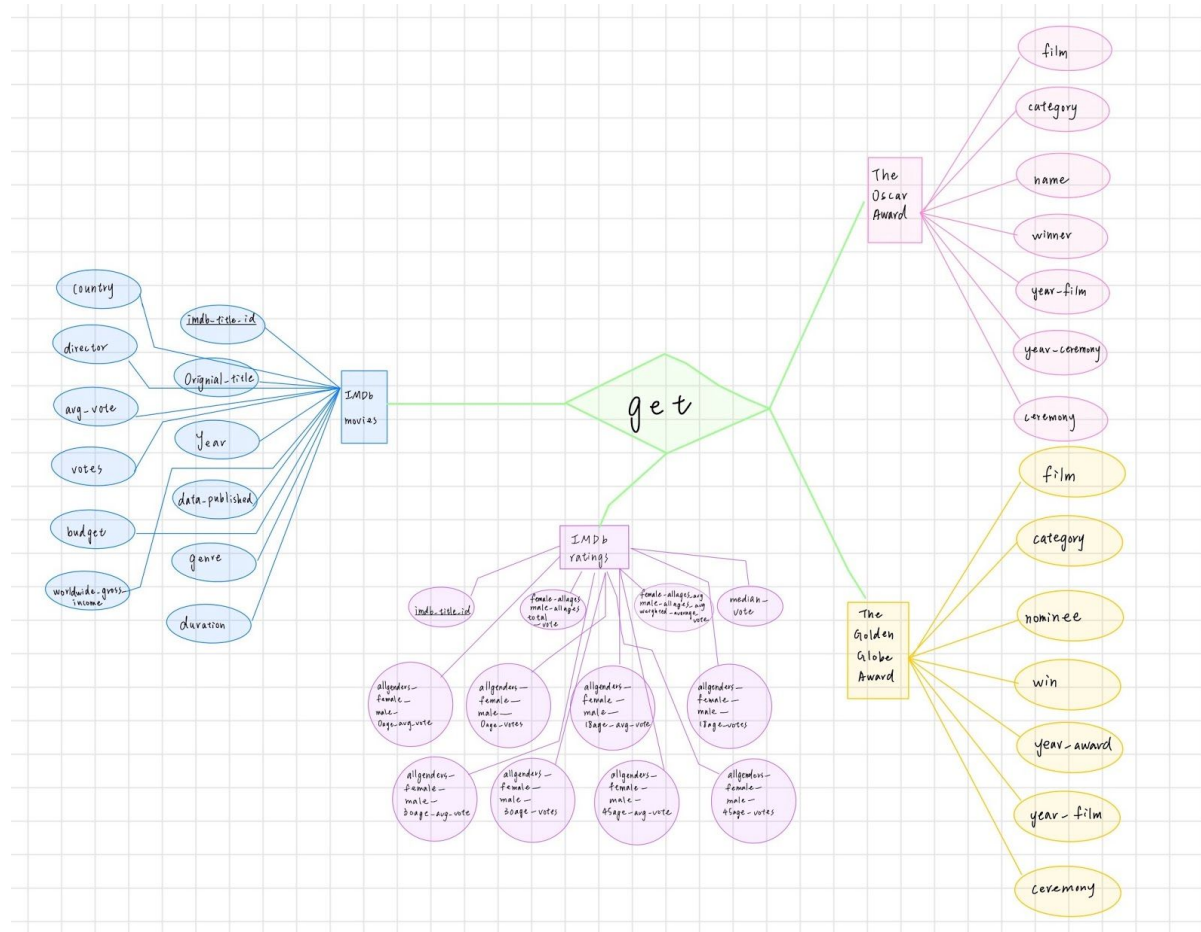
$(\text{sid}, \text{cardno})$  pairs are **redundantly stored**.

- movie、movie\_detail、all\_gender、male、female are in 3NF
  - $\text{id} \rightarrow \text{title}$   
id is a superkey
  - $\text{id} \rightarrow \text{year}$   
id is a superkey
  - $\text{title} \rightarrow \text{year}$   
year is part of some key(id, year) in movie

- (id, title) -> year  
year is part of some key(id, year) in movie
  - (id, year) -> title  
title is part of some key(id, title) in movie
  - (title, year) -> id  
id is a superkey
  - title -> id  
id is a superkey
  - year -> id  
id is a superkey
  - year -> title  
title is part of some key(id, title) in movie
  - ...
- director、genre are in 3NF
  - (director,id)->id  
id is part of key
  - (director,id)->director  
director is part of key
- golden\_globe、oscar are in 3NF
  - (year,nomination,film)->win  
(year,nomination,film) is a superkey.
  - (year,nomination,film)->year  
year is part of key.
  - (year,nomination,film)->nomination  
nomination is part of key.
  - (year,nomination,film)->film  
film is part of key.

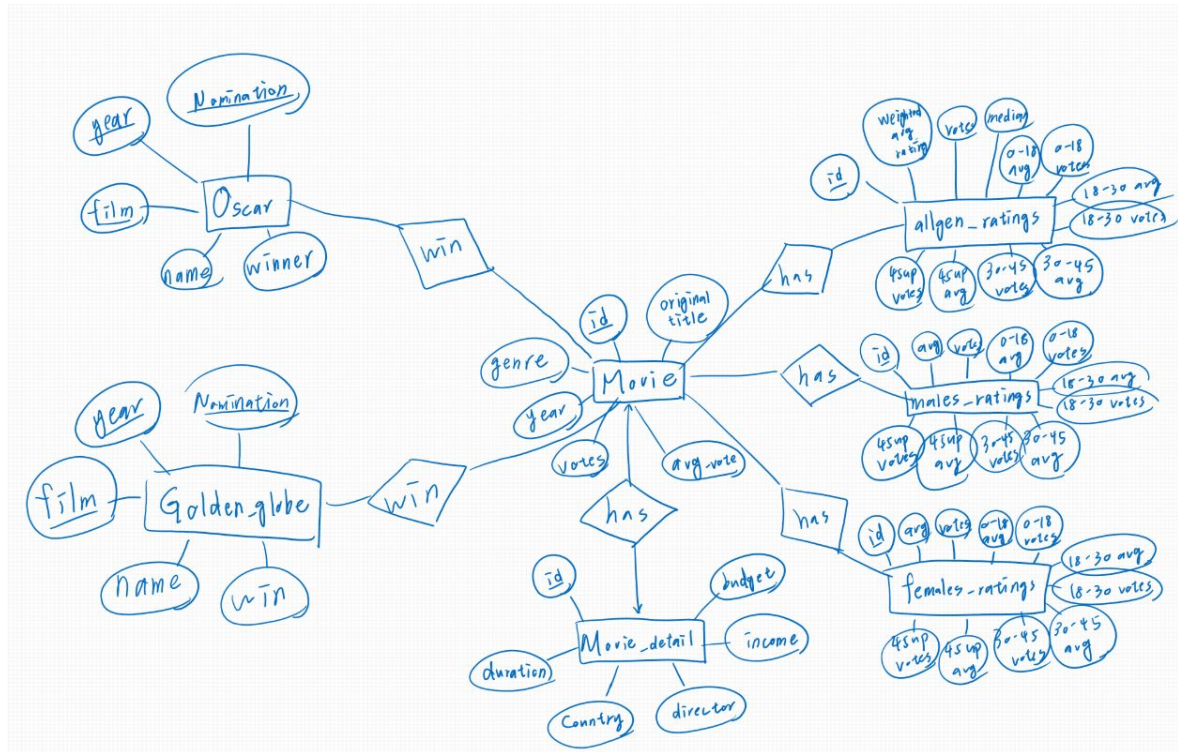
## ● ER Model

- Before  
網路上Dataset最原始的狀況。



○ Middle

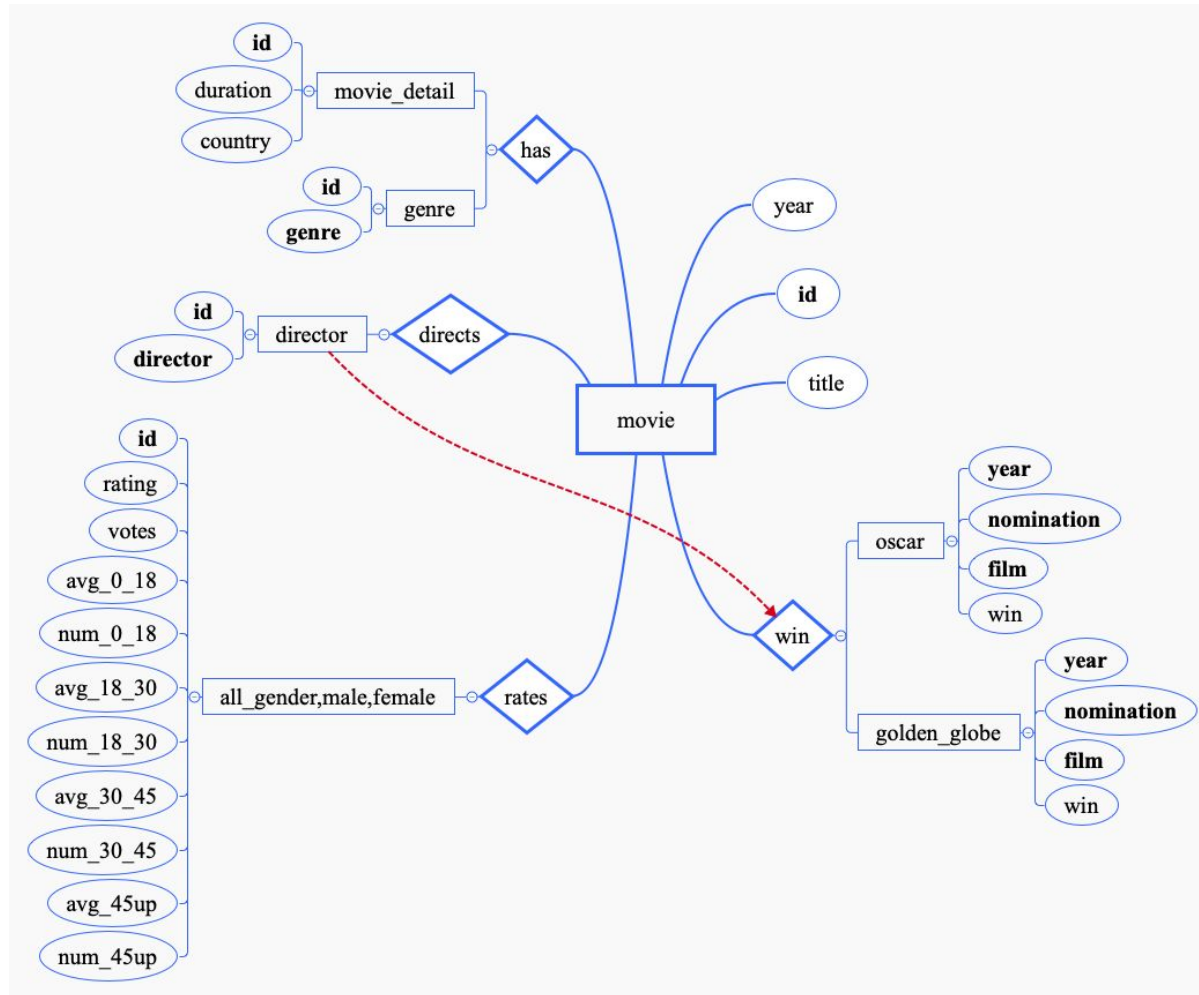
希望能將比較容易一起被用到的element放在同一個table中，一個table中不要包含太多資料。



# ○ After

我們發現每個電影擁有不只一個genre、director，所以我們把genre和director分成另外兩個獨立的table，讓每個電影 id 可以對應到多個genre、director，且讓movie\_detail 中的 id 能成為 primary key。(下圖粗體字代表table中的key)





## Database

- Database

MySQL

- Maintenance

- Update

當使用者輸入關於該電影的評分之後，我們會根據使用者的年齡與性別把相對應的table update。

update公式：

update後評分 = (update前評分 \* update前vote數 + 使用者評分) / update後vote數

update 後vote數 = update前vote數 + 1

```
$sql_up="UPDATE $gender
SET $avg=($avg*$num+$rating)/($num+1), $nums=$num+1, rating=($total_rating*$votes+$rating)/($votes+1), votes=$votes+1
where $gender.id = '$id'";
```

- Insert

在「我要評分」這個功能中，如果使用者想要評分的電影不在database裡面，我們會問使用者要不要新增電影的相關資料（INSERT）。我們會在query design 做更詳細的解釋。

MySQL 的 INSERT query 如以下截圖：

```
$sql='INSERT INTO movie (id,title,year) values ("'. $id. '", "'. $title. '", "'. $year. '" )';
//echo $sql;
```

- Connection Between Database And Application

- Connection

為了讓php連上mysql，需要把預設的驗證方式auth\_socket更改為mysql\_native\_password，使root使用者能利用帳號密碼進入mysql。

```
mysql> select user,authentication_string,plugin,host from mysql.user;
```

user	authentication_string	plugin	host
root		auth_socket	localhost
mysql.session	*THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE	mysql_native_password	localhost
mysql.sys	*THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE	mysql_native_password	localhost
debian-sys-maint	*CE27635178A924C0A836C7CD22BAC7789D496F52	mysql_native_password	localhost

4 rows in set (0.01 sec)

```
mysql> select user,authentication_string,plugin,host from mysql.user;
```

user	authentication_string	plugin	host
root	*F446C894F73A46F460B9D23EF7FAAFB19A14DFE2	mysql_native_password	localhost
mysql.session	*THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE	mysql_native_password	localhost
mysql.sys	*THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE	mysql_native_password	localhost
debian-sys-maint	*CE27635178A924C0A836C7CD22BAC7789D496F52	mysql_native_password	localhost

4 rows in set (0.01 sec)

用new這個運算子建立mysqli這個type的物件存到\$mysqli，使之後的php程式碼可以和這個物件互動。

第一個參數為host name也可以是IP address，指mysql server所在位置；第二個參數為擁有該database權限的username；第三個參數為password；第四個參數為欲使用的database名稱。

成功連上以後就可以使用mysql的database資料執行Query。

```
$mysqli = new mysqli('localhost', 'root', '', 'project');
```

用connect\_errno這個method檢查連線是否成功，如果連線失敗會顯示錯誤訊息並且直接exit。

```
if ($mysqli->connect_errno) {
    echo "Sorry, this website is experiencing problems.";

    echo "Error: Failed to make a MySQL connection, here is why: \n";
    echo "Errno: " . $mysqli->connect_errno . "\n";
    echo "Error: " . $mysqli->connect_error . "\n";

    exit;
}
```

- Query Processing

從網頁的表單拿到使用者選擇的查詢條件。

```
$country=$_POST['country'] ;
$genre= $_POST['genre'];
$rating = $_POST['rating'];
$sort= $_POST['sort'];
$title= $_POST['title'];
$duration= $_POST['duration'];
$rows= $_POST['rows'];
$gender= $_POST['gender'];
$director = $_POST['director'];
$age= $_POST['age'];
$award= $_POST['award'];
$start= $_POST['start'];
$end = $_POST['end'];
$vote_number= $_POST['vote_number'];
$select_in= 'm.title, m.year';
$select_out= 'temp.title, temp.rate, temp.year';
```

檢查使用者輸入的資料是否合法，詳情請見後面Exception Handling。

```
function check($rows,$start,$end,$vote_number)
```

利用if/else條件句，拼湊出符合使用者選擇條件的query。

```
$from=$from.' movie m, all gender al_2';
$where=$where.' m.id=al_2.id ';
$sql_str='SELECT '.$select_out.' FROM'.'(SELECT DISTINCT '.$rating.','.$select_in.' FROM'.$from.' WHERE '.$where.$order.$limit.')as temp';
```

連結mysql database執行query，將執行結果存進\$result，並利用內建methods檢查錯誤，如果query有錯則會印出錯誤訊息並exit。

```

$sql="$sql_str";
if (!$result = $mysqli->query($sql)) {
    // Oh no! The query failed.
    echo "Sorry, the website is experiencing problems.";
    echo "Error: Our query failed to execute and here is why: \n";
    echo "Query: " . $sql . "\n";
    echo "Errno: " . $mysqli->errno . "\n";
    echo "Error: " . $mysqli->error . "\n";
    exit;
}

```

如果查詢結果是empty set，在結果頁面顯示「抱歉,並未查詢到符合條件的電影」。

```

if ($result->num_rows === 0) {
    // Oh, no rows! Sometimes that's expected and okay, sometimes
    // it is not. You decide. In this case, maybe actor_id was too
    // large?
    echo "抱歉，並未查詢到符合條件的電影<br>";
}

```

如果有查詢到相符的資料，就一一將\$result內的資料輸出到結果頁面內的表格。

這裡固定會顯示的是結果行數編號、電影名稱、電影發行年份、電影平均評分，其餘欄位(片長、導演、投票數量)會依據有無選擇限制來顯示。但是國家、類型等等若限制條件則所有查詢結果的該欄位資料都相同，因此不顯示。

```

$title=Title;
$rate=Rating;
echo '<tr><td>','<u>#</u>','</td>';
echo '<td>',$title,'</td>';
echo '<td>','Year','</td>';
echo '<td>',$rate,'</td>';
if($duration!='select')
{
    echo '<td>','Duration','</td>';
}
if($director!='')
{
    echo '<td>','Director','</td>';
}
if($vote_number!='')
{
    echo '<td>','<u># of Votes</u>','</td>';
}
while ($ans = $result->fetch_assoc()) {
    $rowid = $rowid + 1;
    echo '<tr><td>',$rowid,'</td>';
    echo '<td>',$ans['title'],'</td>';
    echo '<td>',$ans['year'],'</td>';
    echo '<td>',$ans['rate'],'</td>';
    if($duration!='select')
    {
        echo '<td>',$ans['duration'],'</td>';
    }
    if($director!='')
    {
        echo '<td>',$ans['director'],'</td>';
    }
}

```

- Exception Handling

- 條件查詢

條件查詢的頁面如下：



Duration Range :

# of Result Rows :  (>0)

Rating of Which Gender :

Director Name Including :

Rating of Which Age :

Award :

Year of Release :  ~

Minimum # of Votes :  (>=0)

根據上圖我們可以知道「條件查詢」可能有四個地方會出錯，分別是# of Result Rows、Year of Release的起訖年（兩格）、Minimum # of Votes。

1. # of Result Rows

若使用者輸入的資料不是數值，會讓query執行出現例外狀況。此條件是讓使用者決定query查詢結果要顯示幾筆資料，如果使用者輸入2.5或“abc”之類的東西，拼湊起來的query（詳情請看前述Query Processing）就會出現“limit abc”

2. Year of Release

若使用者輸入的資料不是數值，會讓query執行出現例外狀況。

3. Minimum # of Votes

若使用者輸入的資料不是數值或是小於0，會讓query執行出現例外狀況。

```

function check($rows,$start,$end,$vote_number)
{
    $error='';
    //error_handling of row
    if(!is_numeric($rows) )
    {
        if($rows!='')$error=$error."顯示結果的數量必須為整數!<br>";
    }
    else
    {
        if($rows <=0)$error=$error."顯示結果的數量必須大於零!<br>";
    }

    if(!is_numeric($start) )
    {
        if($start!='')$error=$error."年份必須為整數!<br>";
    }

    if(!is_numeric($end) )
    {
        if($end!='')$error=$error."年份必須為整數!<br>";
    }

    if(!is_numeric($vote_number) )
    {
        if($vote_number!='')$error=$error."最小投票數量必須為整數!<br>";
    }
    else
    {
        if($vote_number <0) $error=$error."最小投票數量必須大於零!<br>";
    }
    if($error!='')throw new Exception($error);
}

return true;

```

# of Result Rows、Year of Release、Minimum # of Votes若數值不是整數，會讓query執行出現例外狀況，所以若經由前述function檢查通過的話，一律先經過intval()無條件捨去，避免小數的問題。

```
$rows=intval($rows);
$start=intval($start);
$end=intval($end);
$vote_number=intval($vote_number);
```

如果出現起始年大於結束年的情形，顯示警告但不直接結束Query，自動交換起訖年。

```
if($end<$start)
{
    $tem=$start;
    $start=$end;
    $end=$tem;
    echo "Warning:年份範圍輸入不合理";
}
```

實例：release of year：1948~1942

# of Result Rows :  (>0)

Rating of Which Gender :

Director Name Including :

Rating of Which Age :

Award :

Year of Release :  ~

Minimum # of Votes :  (>=0)



顯示結果：

Warning:年份範圍輸入不合理

#	Title	Year	Rating	# of Votes
1	'Gung Ho!': The Story of Carlson's Makin Island Raiders	1943	6.1	106
2	'Neath Brooklyn Bridge	1942	5.8	168
3	13 Rue Madeleine	1946	6.9	251
4	3 Godfathers	1948	7.1	798
5	A Connecticut Yankee in King Arthur's Court	1948	6.6	293
6	A Date with Judy	1948	6.6	271
7	A Double Life	1947	7	345
8	A Foreign Affair	1948	7.4	867
9	A Guy Named Joe	1943	7	353
10	A Lady Takes a Chance	1943	6.5	154

重新查詢

回到主畫面

# of Result Rows、Year of Release、Minimum # of Votes都不是正整數的實例：

# of Result Rows = -2、Year of Release = abc ~ de、Minimum # of Votes = -9

# of Result Rows :  (>0)

Rating of Which Gender :

Director Name Including :

Rating of Which Age :

Award :

Year of Release :  ~

Minimum # of Votes :  (>=0)

顯示結果：

顯示結果的數量必須大於零！  
年份必須為整數！  
年份必須為整數！  
最小投票數量必須大於零！

#### ■ 我要評分

評分前會先要求輸入電影名稱及年份，用來分辨該電影是否已經存在 database 中，如果輸入的電影名稱為空字串或是年份不在 1888~2020 的

範圍內時，會輸出錯誤訊息並請使用者選擇要重新輸入或是直接跳回主畫面。(世界上第一部電影是1888年發行的。)

```
if($title==" "&&($year>2020||$year<1888))
{
    echo "請輸入非空字串的电影名稱<br>";
    echo "請輸入介於1888~2020的年份<br>";
    $action="rate.php";
    $button1="重新輸入";
    $button2="回到主畫面";
}
else if($title==" ")
{
    echo "請輸入非空字串的电影名稱<br>";
    $action="rate.php";
    $button1="重新輸入";
    $button2="回到主畫面";
}
else if($year>2020||$year<1888)
{
    echo "請輸入介於1888~2020的年份<br>";
    $action="rate.php";
    $button1="重新輸入";
    $button2="回到主畫面";
}
```

請輸入非空字串的电影名稱  
請輸入介於1888~2020的年份

重新輸入

回到主畫面

在新增新的電影時，如果片長小於等於零或者導演名稱空白的話會輸出錯誤訊息並請使用者選擇要重新輸入或是回到主畫面。

```
$action=submit;
$action2=hidden;
if($duration<=0)
{
    echo "請輸入>0的影片長度<br>";
}
if($director=="")
{
    echo "請輸入非空字串的導演名稱<br>";
}
```

```
<form action="rate.php" method="post">
<input type="<?php echo $action; ?>" value='重新評分'>
</form>

<form action="info.php" method="post">
<input type='submit' value='回到主畫面'>
</form>
```

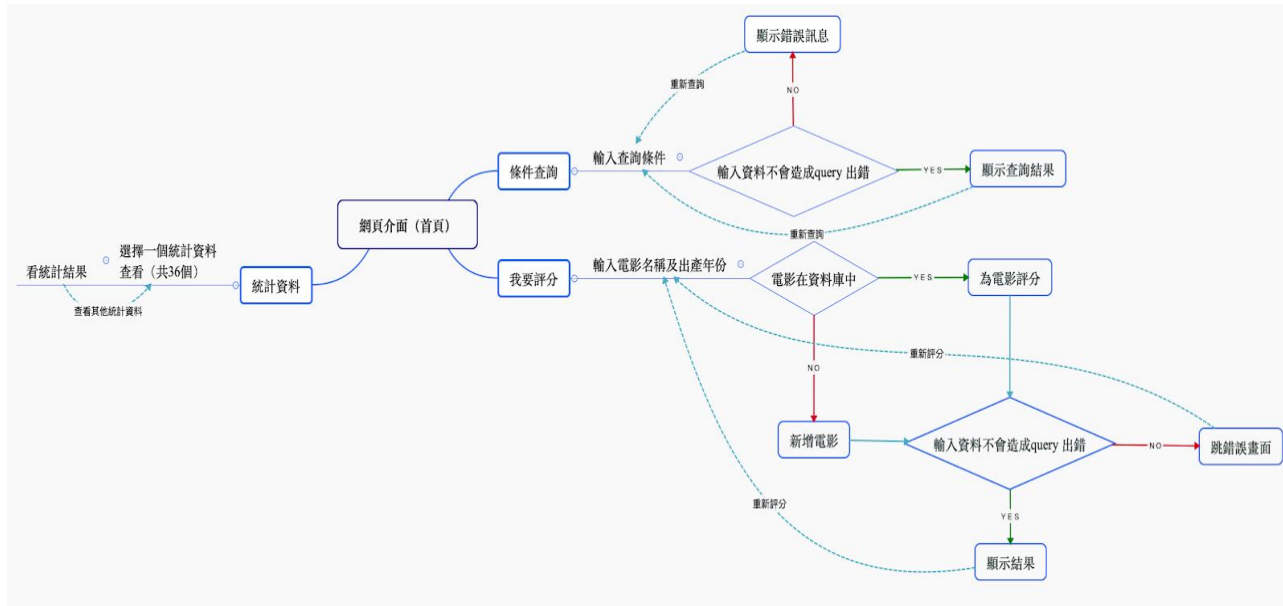
請輸入>0的影片長度  
請輸入非空字串的導演名稱

重新評分

回到主畫面

# Application

- Interface & Functions



- 主畫面：主畫面我們提供使用者三種不同種類功能



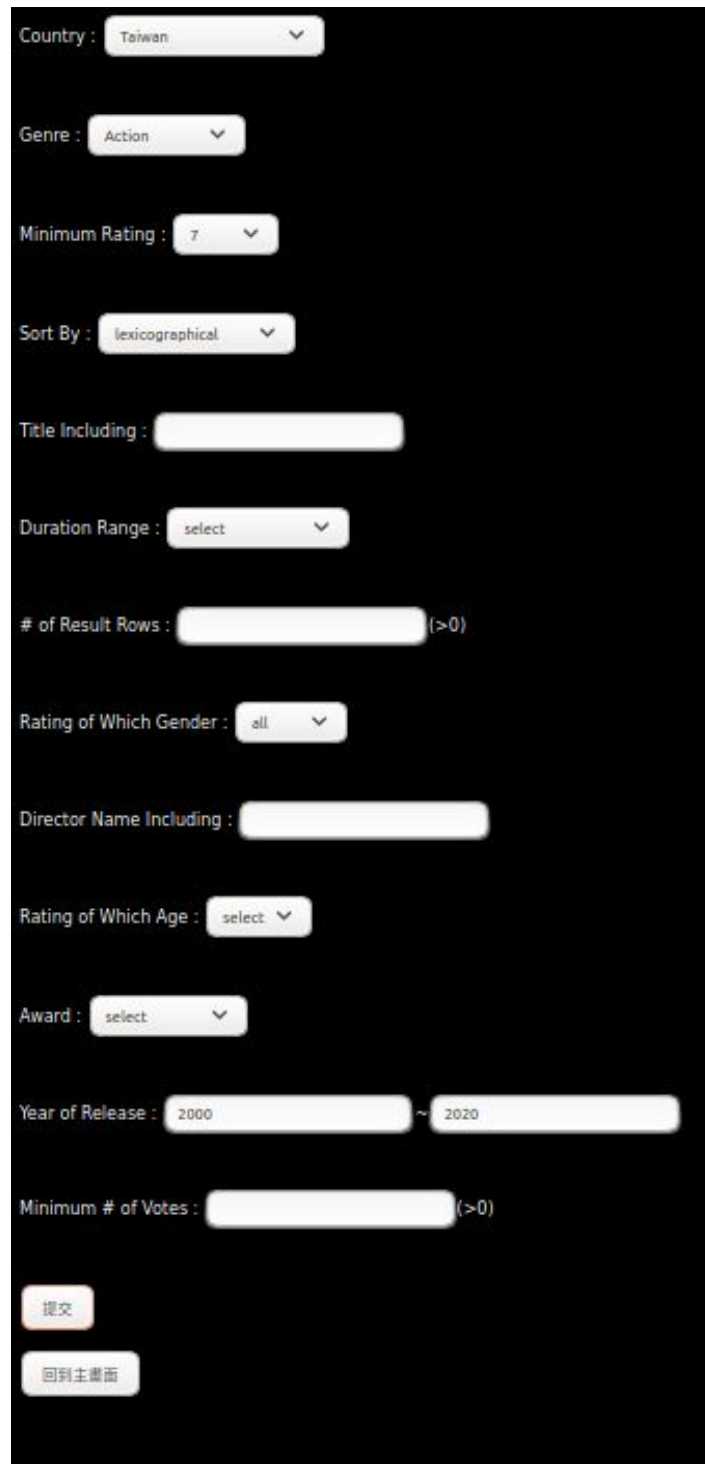
- 統計數據：點入統計數據這個頁面後，會有36個已經寫好的Query主題提供使用者查看有趣的數據分析



- 條件查詢：點入條件查詢之後，使用者會看見一個查詢表單，使用者依照他們想要限定的條件輸入到查詢表單(不一定所有的條件都要選擇，可以放著不動)，輸入完畢後按提交鍵，接著使用者便會得到依照之前所選條件篩選出來的結果。條件選項詳細說明如下：

- Country：下拉式選單，電影出產國家，預設為不限
- Genre：下拉式選單，電影類型，預設為不限
- Minimum Rating：下拉式選單，輸出結果的最低評分門檻，0~10，預設為0
- Sort By：下拉式選單，字典序(預設)、評分低到高、評分高到低
- Title Including：手動輸入，標題必須包含的字元(模糊查詢)，可輸入萬用符號、大小寫不分，預設為不限
- Duration Range：下拉式選單，片長小於90分鐘、90~150分鐘、大於150分鐘，預設為不限
- # of Result Rows：手動輸入，欲顯示的查詢結果行數，預設為不限
- Rating of Which Gender：下拉式選單，欲察看的評分來源，有男性、女性、全部(預設)
- Director Name Including：手動輸入，導演姓名必須包含的字元(模糊查詢)，可輸入萬用符號、大小寫不分，預設為不限

- Rating of Which Age：下拉式選單，欲察看的評分來源，有0~17歲、18~29歲、30~44歲、45歲以上，預設為不限
- Award：下拉式選單，篩選出得過特殊獎項的電影，金球獎、奧斯卡獎，預設為不限
- Year of Release：手動輸入，電影發行日範圍，起始年份、結束年份，預設為不限
- Minimum # of Votes：手動輸入，最低投票數量門檻，預設為不限



Country : Taiwan

Genre : Action

Minimum Rating : 7

Sort By : lexicographical

Title Including :

Duration Range : select

# of Result Rows : (>0)

Rating of Which Gender : all

Director Name Including :

Rating of Which Age : select

Award : select

Year of Release : 2000 ~ 2020

Minimum # of Votes : (>0)

提交

回到主畫面



#	Title	Year	Rating
1	Kuang tu	2018	7.3
2	Sai de ke · ba lai: Cai hong qiao	2011	7.8
3	Sai de ke · ba lai: Tai yang qi	2011	7.8

重新查詢

回到主畫面

- 我要評分：點入我要評分後，會出現兩個輸入欄位，第一個欄位輸入使用者要評分的電影名稱(完整查詢，但大小寫不分)，第二個欄位則是輸入電影的發行年分(避免出現同名的電影)，輸入完成後按提交鍵。

Title : The Story of the Kelly Gang

Year : 1906

提交

回到主畫面

Title : db project

Year : 2020

提交

回到主畫面

- 提交：讓使用者在評分前再次確定要評分的電影，並按「我要評分」進入評分階段，評分階段分為為舊電影評分及新增新電影並為其評分，或是按「算了,我累了」回到主畫面。

是否為年份為1906的The Story of the Kelly Gang 評分

我要評分

算了,我累了

這部電影不在database中,是否新增電影並為其評分

我要評分

算了,我累了

■ 為舊電影評分：

- 輸入使用者的年齡範圍、性別、對選定電影的評分(1~10)，並按下提交鍵更新資料庫。





rating : 1 ▾

gender : male ▾

age: 0-18 ▾

提交

回到主畫面

- 提交：按下提交鍵後，使用者能選擇是否查看評分後該電影的詳細資料。



評分完成

顯示結果

回到主畫面

- 顯示結果：按下顯示結果後會出現update後的資料

### Movie

id	title	year
tt0000574	The Story of the Kelly Gang	1906

### Movie\_Detail

id	duration	country
tt0000574	70	Australia

### Director

id	director
tt0000574	Charles Tait

### Genre

id	genre
tt0000574	Biography
tt0000574	Crime
tt0000574	Drama

### All\_Gender

id	rating	votes	avg_0_18	num_0_18	avg_18_30	num_18_30	avg_30_45	num_30_45	avg_45up	num_45up
tt0000574	6.09796	538	6	1	6.2	126	5.89571	210	6.4	100

### Male

id	rating	votes	avg_0_18	num_0_18	avg_18_30	num_18_30	avg_30_45	num_30_45	avg_45up	num_45up
tt0000574	6.1	393	6	1	6.2	112	5.9	186	6.4	85

### Female

id	rating	votes	avg_0_18	num_0_18	avg_18_30	num_18_30	avg_30_45	num_30_45	avg_45up	num_45up
tt0000574	6.07755	49	0	0	5.7	14	5.95	20	6.6	14

重新評分

回到主畫面

- 新增新電影並為其評分：
  - 輸入拍攝該電影的國家、導演，該電影的類型、片長，使用者的年齡範圍、性別、對選定電影的評分(1~10)，並按下提交鍵對資料庫新增資料。

country : Taiwan

genre : Drama

rating : 6

duration : 99

gender : female

director: me

age: 18-30

提交

回到主畫面

- 提交：按下提交鍵後，使用者能選擇是否查看評分後該電影的詳細資料。

評分完成

顯示結果

回到主畫面

- 顯示結果：按下顯示結果後會出現新電影評分的資料

**Movie**

id	title	year
new81273	db project	2020

**Movie Detail**

id	duration	country
new81273	99	Taiwan

**Director**

id	director
new81273	me

**Genre**

id	genre
new81273	Drama

**All Gender**

id	rating	votes	avg_0_18	num_0_18	avg_18_30	num_18_30	avg_30_45	num_30_45	avg_45up	num_45up
new81273	6	1	0	0	6	1	0	0	0	0

**Male**

id	rating	votes	avg_0_18	num_0_18	avg_18_30	num_18_30	avg_30_45	num_30_45	avg_45up	num_45up
new81273	0	0	0	0	0	0	0	0	0	0

**Female**

id	rating	votes	avg_0_18	num_0_18	avg_18_30	num_18_30	avg_30_45	num_30_45	avg_45up	num_45up
new81273	6	1	0	0	6	1	0	0	0	0

重新評分

回到主畫面

- Query Design & Exception Handling

- Query Design

- search

- 利用多個if/else statement(如下圖)來建造出符合使用者搜尋條件的query，在每個if/else statement 中將該個條件需要的table append到string(\$from)中並把篩選條件append到string(\$where)中，最後再將 使用者輸入的row個數，和選擇的希望的排列方法分別集合 LIMIT 和ORDER BY 來決定query 呈現的方式。
      - 我們之所以這樣設計我們的query是因為我們不希望一開始就把所有database裡的table 都append 到 \$from中，利用if/else statement 我們只選需要用到的table，希望這樣能減少join 的table數量。
      - 篩選電影製造country

```
if($country!='select')
{
    $from=' movie_detail c,';
    $where='m.id=c.id and c.country='.$country.' and ';
}
```

- 篩選電影genre

```
if($genre!='select')
{
    $from=$from.' genre g,';
    $where=$where.' g.id=m.id and g.genre='.$genre.' and ';
}
```

- 依照使用者希望的篩選rating依據的性別和年齡來篩選電影的 rating-1

```
if($rating!='select')
{
    //echo "$gender/n";
    if($gender=='all')
    {
        if($age=='select')
        {
            $from=$from.' all_gender al, ';
            $where=$where.' al.id=m.id and al.rating>='.$rating.' and ';
            $order=' al.rating';
        }
        else if($age=='0_18')
        {
            $from=$from.' all_gender al, ';
            $where=$where.' al.id=m.id and al.avg_0_18>='.$rating.' and ';
            $order='al.avg_0_18';
        }
        else if($age=='18_30')
        {
            $from=$from.' all_gender al, ';
            $where=$where.' al.id=m.id and al.avg_18_30>='.$rating.' and ';
            $order='al.avg_18_30';
        }
        else if($age=='30_45')
        {
            $from=$from.' all_gender al, ';
            $where=$where.' al.id=m.id and al.avg_30_45>='.$rating.' and ';
            $order='al.avg_30_45';
        }
        else
        {
            $from=$from.' all_gender al, ';
            $where=$where.' al.id=m.id and al.avg_45up>='.$rating.' and ';
            $order='al.avg_45up';
        }
    }
}
```

- 依照使用者希望的篩選rating依據的性別和年齡來篩選電影的 rating-2

```

else if($gender=='male')
{
    if($age=='select')
    {
        $from=$from.' male al, ';
        $where=$where.' al.id=m.id and al.rating>= '.$rating.' and ';
        $order='al.rating';
    }
    else if($age=='0_18')
    {
        $from=$from.' male al, ';
        $where=$where.' al.id=m.id and al.avg_0_18>= '.$rating.' and ';
        $order='al.avg_0_18';
    }
    else if($age=='18_30')
    {
        $from=$from.' male al, ';
        $where=$where.' al.id=m.id and al.avg_18_30>= '.$rating.' and ';
        $order='al.avg_18_30';
    }
    else if($age=='30_45')
    {
        $from=$from.' male al, ';
        $where=$where.' al.id=m.id and al.avg_30_45>= '.$rating.' and ';
        $order='al.avg_30_45';
    }
    else
    {
        $from=$from.' male al, ';
        $where=$where.' al.id=m.id and al.avg_45up>= '.$rating.' and ';
        $order='al.avg_45up';
    }
}
}

```

- 依照使用者希望的篩選rating依據的性別和年齡來篩選電影的 rating-3

```

else
{
    if($age=='select')
    {
        $from=$from.' female al, ';
        $where=$where.' al.id=m.id and al.rating>= '.$rating.' and ';
        $order='al.rating';
    }
    else if($age=='0_18')
    {
        $from=$from.' female al, ';
        $where=$where.' al.id=m.id and al.avg_0_18>= '.$rating.' and ';
        $order='al.avg_0_18';
    }
    else if($age=='18_30')
    {
        $from=$from.' female al, ';
        $where=$where.' al.id=m.id and al.avg_18_30>= '.$rating.' and ';
        $order='al.avg_18_30';
    }
    else if($age=='30_45')
    {
        $from=$from.' female al, ';
        $where=$where.' al.id=m.id and al.avg_30_45>= '.$rating.' and ';
        $order='al.avg_30_45';
    }
    else
    {
        $from=$from.' female al, ';
        $where=$where.' al.id=m.id and al.avg_45up>= '.$rating.' and ';
        $order='al.avg_45up';
    }
}
}

```

- 依照使用者希望的篩選rating依據的性別和年齡來篩選電影的 rating-4

```

else
{
    $order=' al_2.rating ';
}

```

- 篩選title裡面需要包含的string(利用LIKE ,%)

```

if($title!='')
{
    $from=$from.' movie m1, ';
    $where=$where.' m1.id=m.id and m1.title like '.$title.'%' and ';
}

```

- 篩選duration的長度

```

if($duration!='select')
{
    //echo "$gender/n";
    if($duration=='lower_than_90')
    {
        $from=$from.' movie_detail m1_1, ';
        $where=$where.' m1_1.id=m.id and m1_1.duration <90 and ';
        $select_in=$select_in.', m1_1.duration';
        $select_out=$select_out.', temp.duration';
    }
    else if($duration=='90_to_150')
    {
        $from=$from.' movie_detail m1_1, ';
        $where=$where.' m1_1.id=m.id and m1_1.duration >90 and m1_1.duration<150 and ';
        $select_in=$select_in.', m1_1.duration';
        $select_out=$select_out.', temp.duration';
    }
    else
    {
        $from=$from.' movie_detail m1_1, ';
        $where=$where.' m1_1.id=m.id and m1_1.duration >150 and ';
        $select_in=$select_in.', m1_1.duration';
        $select_out=$select_out.', temp.duration';
    }
}

```

- 篩選director裡面需要包含的string(利用LIKE ,%)

```

if($director!='')
{
    $from=$from.' director d, ';
    $where=$where.' d.id=m.id and d.director like '.$director.'%' and ';
    $select_in=$select_in.', d.director';
    $select_out=$select_out.', temp.director';
}
if($award!='select')
{
    $from=$from.' '. $award.' a, ';
    $where=$where.' a.film=m.title and a.win=\TRUE\' and ';
}
if($start!='&& $end!='')
{
    $where=$where.' m.year >= '.$start.' and m.year<= '.$end.' and ';
}

```

- 依照使用者希望的篩選vote number依據的性別和年齡來篩選電影的vote number-1



```

if($vote_number!='')
{
    //echo "$gender/n";
    if($gender=='all')
    {
        if($age=='select')
        {
            $from=$from.' all_gender al_1, ';
            $where=$where.' al_1.id=m.id and al_1.votes> '.$vote_number.' and ';
            $select_in=$select_in.', al_1.votes as vote_number';
            $select_out=$select_out.', temp.vote_number';
        }
        else if($age=='0_18')
        {
            $from=$from.' all_gender al_1, ';
            $where=$where.' al_1.id=m.id and al_1.num_0_18> '.$vote_number.' and ';
            $select_in=$select_in.', al_1.num_0_18 as vote_number';
            $select_out=$select_out.', temp.vote_number';
        }
        else if($age=='18_30')
        {
            $from=$from.' all_gender al_1, ';
            $where=$where.' al_1.id=m.id and al_1.num_18_30> '.$vote_number.' and ';
            $select_in=$select_in.', al_1.num_18_30 as vote_number';
            $select_out=$select_out.', temp.vote_number';
        }
        else if($age=='30_45')
        {
            $from=$from.' all_gender al_1, ';
            $where=$where.' al_1.id=m.id and al_1.num_30_45> '.$vote_number.' and ';
            $select_in=$select_in.', al_1.num_30_45 as vote_number';
            $select_out=$select_out.', temp.vote_number';
        }
    }
}

```

- 依照使用者希望的篩選vote number依據的性別和年齡來篩選電影的vote number-2

```

    else
    {
        $from=$from.' all_gender al_1, ';
        $where=$where.' al_1.id=m.id and al_1.num_45up> '.$vote_number.' and ';
        $select_in=$select_in.', al_1.num_45up as vote_number';
        $select_out=$select_out.', temp.vote_number';
    }
}
else if($gender=='male')
{
    if($age=='select')
    {
        $from=$from.' male al_1, ';
        $where=$where.' al_1.id=m.id and al_1.votes> '.$vote_number.' and ';
        $select_in=$select_in.', al_1.votes as vote_number';
        $select_out=$select_out.', temp.vote_number';
    }
    else if($age=='0_18')
    {
        $from=$from.' male al_1, ';
        $where=$where.' al_1.id=m.id and al_1.num_0_18> '.$vote_number.' and ';
        $select_in=$select_in.', al_1.num_0_18 as vote_number';
        $select_out=$select_out.', temp.vote_number';
    }
    else if($age=='18_30')
    {
        $from=$from.' male al_1, ';
        $where=$where.' al_1.id=m.id and al_1.num_18_30> '.$vote_number.' and ';
        $select_in=$select_in.', al_1.num_18_30 as vote_number';
        $select_out=$select_out.', temp.vote_number';
    }
    else if($age=='30_45')
    {
        $from=$from.' male al_1, ';
        $where=$where.' al_1.id=m.id and al_1.num_30_45> '.$vote_number.' and ';
        $select_in=$select_in.', al_1.num_30_45 as vote_number';
        $select_out=$select_out.', temp.vote_number';
    }
}
}

```

- 依照使用者希望的篩選vote number依據的性別和年齡來篩選電影的vote number-3

```

    else
    {
        $from=$from.' male al_1, ';
        $where=$where.' al_1.id=m.id and al_1.num_45up> '.$vote_number.' and ';
        $select_in=$select_in.', al_1.num_45up as vote_number';
        $select_out=$select_out.', temp.vote_number';
    }
}
else
{
    if($age=='select')
    {
        $from=$from.' female al_1, ';
        $where=$where.' al_1.id=m.id and al_1.votes> '.$vote_number.' and ';
        $select_in=$select_in.', al_1.votes as vote_number';
        $select_out=$select_out.', temp.vote_number';
    }
    else if($age=='0_18')
    {
        $from=$from.' female al_1, ';
        $where=$where.' al_1.id=m.id and al_1.num_0_18> '.$vote_number.' and ';
        $select_in=$select_in.', al_1.num_0_18 as vote_number';
        $select_out=$select_out.', temp.vote_number';
    }
    else if($age=='18_30')
    {
        $from=$from.' female al_1, ';
        $where=$where.' al_1.id=m.id and al_1.num_18_30> '.$vote_number.' and ';
        $select_in=$select_in.', al_1.num_18_30 as vote_number';
        $select_out=$select_out.', temp.vote_number';
    }
}
}

```

- 依照使用者希望的篩選vote number依據的性別和年齡來篩選電影的vote number-4

```

    else if($age=='30_45')
    {
        $from=$from.' female al_1, ';
        $where=$where.' al_1.id=m.id and al_1.num_30_45> '.$vote_number.' and ';
        $select_in=$select_in.', al_1.num_30_45 as vote_number';
        $select_out=$select_out.', temp.vote_number';
    }
    else
    {
        $from=$from.' female al_1, ';
        $where=$where.' al_1.id=m.id and al_1.num_45up> '.$vote_number.' and ';
        $select_in=$select_in.', al_1.num_45up as vote_number';
        $select_out=$select_out.', temp.vote_number';
    }
}

```

- 根據使用者的選擇來決定query output的方式

```

$rating=$order.' as rate';
if($sort=='rating_low_to_high')
{
    $order=' ORDER BY '.$order.' ASC ';
}
else if($sort=='rating_high_to_low')
{
    $order=' ORDER BY '.$order.' DESC ';
}
else
{
    $order=' ORDER BY m.title ';
}
if($rows!='')
{
    $limit=' LIMIT '.$rows;
}
else
{
    $limit='';
}

```

- 組合query

```

$from=$from.' movie m, all_gender al_2';
$where=$where.' m.id=al_2.id ';
$sql_str='SELECT '.$select_out.' FROM'.'(SELECT DISTINCT '.$rating.', '.$select_in.' FROM'.$from.' WHERE '.$where.$order.$limit.')as temp';

```

- 「我要評分」的電影沒出現在database (insert)：



- 首先，先給新電影一個id，id的產生方法是“new”+現在movie 這個table有多少row。
- 會這樣命名是因為我們沒有delete的功能，所以row的數量只增不減，不會有id重複的情形發生（原本的id是“tt”+7個數字，使用者新增的電影是new開頭）。

```
$sql='SELECT movie.id from movie;';
$num_row=mysqli->query($sql)->num_rows;
//echo "rows:". $num_row;
$id="new".$num_row;
//echo "ID:". $id;
//error handling
```

- 把使用者輸入的資料insert到movie、movie\_detail、genre、director這四個table中

```
$sql='INSERT INTO movie (id,title,year) values ("'.$id.'","'.$title.'","'.$year.'")';
//echo $sql;
```

```
$sql='INSERT INTO movie_detail (id,duration,country) values ("'.$id.'","'.$duration.'","'.$country.'")';
// echo $sql;
```

```
$sql='INSERT INTO genre (id,genre) values ("'.$id.'","'.$genre.'")';
//echo $sql;
```

```
$sql='INSERT INTO director (id,director) values ("'.$id.'","'.$director.'")';
//echo $sql;
```

- 把新電影的id還有初始化的值(0)insert 至 all\_gender、male、female這三個table中。

```
$sql='INSERT INTO all_gender (id,rating,votes,avg_0_10,num_0_10,avg_10_30,num_10_30,avg_30_45,num_30_45,avg_45up,num_45up) values ("'.$id.'",0,0,0,0,0,0,0,0,0,0)';
// echo $sql;
```

```
$sql='INSERT INTO female (id,rating,votes,avg_0_10,num_0_10,avg_10_30,num_10_30,avg_30_45,num_30_45,avg_45up,num_45up) values ("'.$id.'",0,0,0,0,0,0,0,0,0,0)';
//echo $sql;
```

- 針對使用者輸入的年齡跟性別去update all\_gender、(male/female)這兩個table。

```

if($age<18)
{
    $column_avg='avg_0_18';
    $column_vote='num_0_18';
}
else if($age<30)
{
    $column_avg='avg_18_30';
    $column_vote='num_18_30';
}
else if($age<45)
{
    $column_avg='avg_30_45';
    $column_vote='num_30_45';
}
else
{
    $column_avg='avg_45up';
    $column_vote='num_45up';
}
//echo "$column_avg $column_vote";

```

```

$sql='UPDATE all_gender SET rating='.$rating.',votes=1,'.$column_avg.'='.$rating.', '.$column_vote.'=1 where all_gender.id="'.$id.'";';

```

```

$sql='UPDATE '.$gender.' SET rating='.$rating.',votes=1,'.$column_avg.'='.$rating.', '.$column_vote.'=1 where '.$gender.'.id="'.$id.'";';
//echo $sql;

```

- 實例：
  - 下圖是使用者輸入的新電影資料

country : Taiwan ▾

genre : Drama ▾

rating : 6 ▾

duration : 99

gender : female ▾

director: me

age: 18-30 ▾

提交

回到主畫面

- insert完之後，新電影在所有的table中的資料如下：

**Movie**

id	title	year
new81273	db project	2020

**Movie Detail**

id	duration	country
new81273	99	Taiwan

**Director**

id	director
new81273	me

**Genre**

id	genre
new81273	Drama

**All\_Gender**

id	rating	votes	avg_0_18	num_0_18	avg_18_30	num_18_30	avg_30_45	num_30_45	avg_45up	num_45up
new81273	6	1	0	0	6	1	0	0	0	0

**Male**

id	rating	votes	avg_0_18	num_0_18	avg_18_30	num_18_30	avg_30_45	num_30_45	avg_45up	num_45up
new81273	0	0	0	0	0	0	0	0	0	0

**Female**

id	rating	votes	avg_0_18	num_0_18	avg_18_30	num_18_30	avg_30_45	num_30_45	avg_45up	num_45up
new81273	6	1	0	0	6	1	0	0	0	0

重新評分

回到主畫面

- insert這個function之所以會這樣設計，是為了符合方便簡約易懂的概念。在設計這個function的時候原本有考慮要用if/else判斷年齡層及性別一次把所有的值insert完（用mysql中的insert\*3+if/else statement\*10(用{}算)），但後來上網搜尋發現mysql update的cost不高，所以就決定用現在這個寫法了（mysql中的insert\*3+mysql中的update\*2+if/else statement\*4(用{}算)）。
- Exception Handling(詳情請見前面Database - Exception Handling)
  - 當使用者輸入不符合格式的資訊時會跳出錯誤訊息，並且可讓使用者選擇要重新查詢或跳回主畫面

顯示結果的數量必須大於零！  
 年份必須為整數！  
 年份必須為整數！  
 最小投票數量必須大於零！

重新查詢

回到主畫面

Country:

Genre:

Minimum Rating:

Sort By:

Title Including:

Duration Range:

# of Result Rows:  (>0)

Rating of Which Gender:

Director Name Including:

Rating of Which Age:

Award:

Year of Release:  -

Minimum # of Votes:  (>=0)

- 當使用者輸入不符合格式的資訊時會跳出錯誤訊息，並且可讓使用者選擇要重新輸入或跳回主畫面

請輸入非空字串的電影名稱

請輸入介於1888~2020的年份

Title:

Year:

- 當使用者輸入不符合格式的資訊時會跳出錯誤訊息，並且可讓使用者選擇要重新評分或跳回主畫面

請輸入>0的影片長度  
請輸入非空字串的導演名稱

Title:

Year:

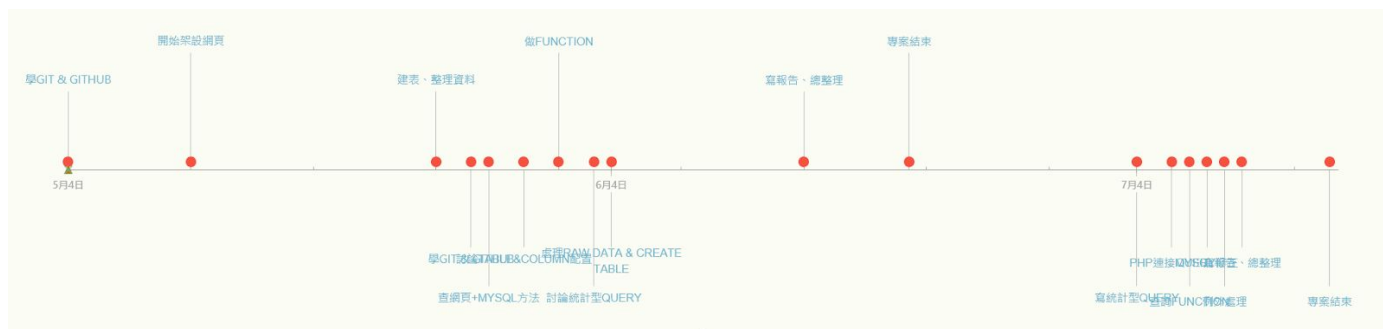
## Others

- Progress

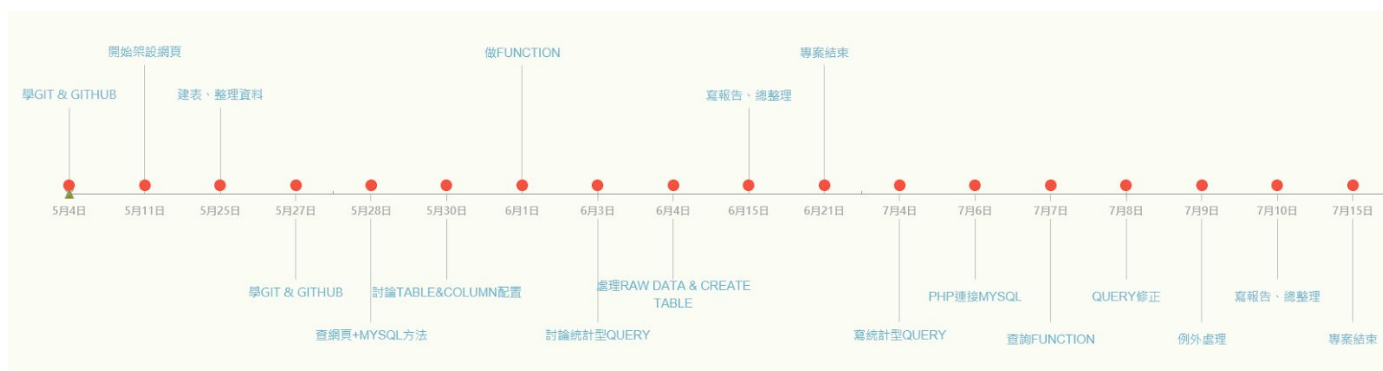
- 時程差異

原先預計每個禮拜做一點事情，沒想到學期中居然有這麼多作業考試，導致中間6月完全在寫作業和考期末考。但最後交出去HW4之後，進度突飛猛進，原先以為只能交半成品的我們，就這麼順利完成了！

- 實際時間軸長相



## ● 詳細日期和進度



## ○ 內容差異

- 哪個導演競逐獎項的時間最久? 這個query並未完成。
- 使用git時，原本預計每個組員都開一個branch，將自己寫的部分丟到自己的branch，但最後全部都丟在master並使用資料夾分開。
- 在呈現資料時原先想讓使用者選擇要呈現出的column，但後來此功能並未實現。
- 原本預期的exception handling 只有當使用者想要將評分打超過10分或是負分時要做處理，最後呈現時將查詢部分的country, rating, genre, sort, duration, gender, age, award都用下拉式選單，避免使用者輸入預期外的資訊。輸入投票最低門檻數量小於零的話，會跳出錯誤資訊並要求重新查詢或回到主畫面。評分時若有要輸入的部分都有做例外處理，像是電影名稱空白，上映年分並不在1888~2020，片長小於零的情況都會跳出錯誤訊息並要求重新評分或回到主畫面。

## ● Problems & Solutions

Q: 原始的csv檔的投票數量有許多空白

A: 用excel將所有空白的地方選取出來後一次補零

Q: 原始的csv檔有許多空字串

A: 在load資料時用delete將該資料刪除

Q: 原始的csv檔有些資料會位於同一欄(電影類型、導演和國家)，中間使用逗號分隔，造成query不好寫，無法使用GROUP BY

A: 電影類型拆成三個.csv檔，導演拆成兩個csv檔，國家只留下第一個國家

Q: 原始的csv檔有些得獎的電影名稱是空白

A: 發現這種的都是跟前一欄資料相同才這樣，所以我們手動複製貼上處理

Q:一開始無法使用mysql -u root -p進入mysql，必須要 sudo mysql -u root -p才可以，因此php無法順利連上mysql

A: 驗證方式由auth\_socket更改為 mysql\_native\_password

Q:原本顯示資料時只有將各結果中間以空白隔開，造成閱讀的不方便

A:用表格呈現及加上column name與rowid

Q:同一個網頁用多個表格來呈現多個query的結果時無法寫在同一個<?php ?>內

A: 因為<?php><?>必須寫在<table></table>裡面，因此分成很多個<table></table>裡面在含<?php ?> 即可

Q: 在一串string後加."\\n"顯示出來的結果不會換行

A:後來發現."\\n"是搭配 echo用才有效，如果要在沒有echo的情形下換行要加<br>

Q: 當要評分時，會分成舊電影評分及新增新電影評分，兩個需要跳轉到的頁面不同，button上所需顯示的訊息也不同

A: 用<?php echo \$xxx?>來決定要跳往哪個畫面及顯示的訊息

Q: 有些頁面需要用到上上頁使用者輸入的資訊，但在上一頁不須選擇

A: 在上一頁將input type改成hidden，submit後在該頁面就可以拿到所需的值

Q:有些在「統計數據」的query跑很久

A:後來發現那些跑很久的table有一次join三個table的情形(執行時間 $O(n^3)$ ，n是table有多少rows)，所以把query改成一次最多join兩個table這個問題就解決了(執行時間 $O(n^2)$ ，n是table有多少rows)。

Q:exception handling判斷使用者輸入是否為整數時無法用is\_int，因為is\_int無法判斷string中的整數

A:改用is\_numeric，但小數也符合is\_numeric因此再將其轉為int

## ● Contribution of Each Team Member

- 0716002 李昕穎：寫create table/load data、撰寫statistics query、dynamic query、data update，和寫report
- 0716008 呂沛儒：寫create table/load data、寫statistics query、改data update（我要評分電影有在database的情形）、寫add new data（我要評分電影不在database的情形）、加條件查詢的error handling和寫報告
- 0716080 劉文心：初期擔任組長督促進度和分配工作、git(強迫)使用教學、寫create table/load data、寫統計型query、優化網頁可讀性、新增rowid、檢查query是否錯誤或尚可優化、使查詢結果動態顯示欄位、報告編修
- 0716203 黃子潔：寫create table/load data、將其他組員寫好的query結果呈現在網頁上，網頁跳轉、下拉式選單及輸入資訊，我要評分的error handling和寫報告



- Repository

#### GitHub

最終版本php資料夾名稱：final\_php

最終版本create database資料夾名稱：final\_create\_tables

- Discussion Channel

#### HackMD