SQL 的五十道練習

> *垂直與水平合併*

數據交點 | 郭耀仁 yaojenkuo@datainpoint.com

這個章節要學起來的 SQL 保留字

- `UNION`
- `ALL`
- `JOIN`
- `ON`
- `LEFT JOIN`

關聯資料表

在「子查詢 」章節中，我們知道子查詢常見的應用情境有：

- 查詢的篩選條件必須要先做一個查詢才能得知。
- 查詢的計算內容必須要先做一個查詢才能得知。
- 查詢所需要的**資料來自不同資料表**。

回憶一下當時如何回答這個問題：我們想知道在 `imdb` 資料庫中，哪幾部電影 Tom Hanks 有演出？

- 先從 `actors` 資料表查詢 Tom Hanks 的演員編號是多少。
- 再依據前一個查詢結果去 `casting` 資料表查詢。
- 再依據前一個查詢結果去 `movies` 資料表查詢。

這意味著我們的查詢範圍開始由「單個」資料表擴展至「多個」資料表

- 子查詢。
- 垂直與水平合併。

關聯式資料庫（Relational Database）的核心精神

- 可以設計資料表彼此之間的「關聯」特性。
- 讓資料表彼此之間的重複性、重疊性降低。

資料表能夠從兩個維度關聯

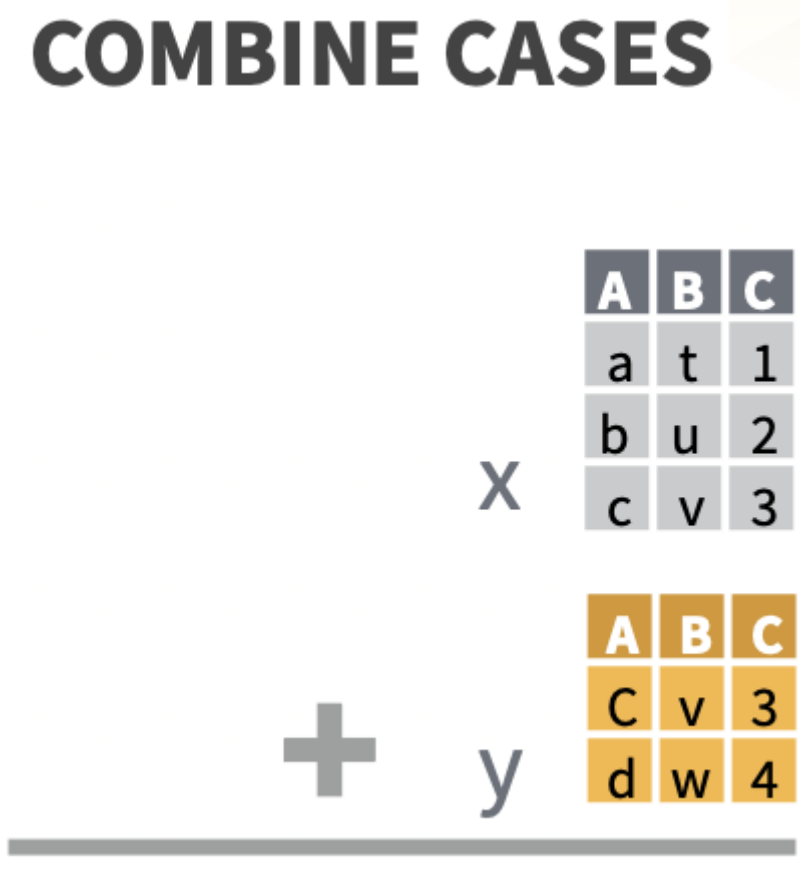
- 垂直合併：讓資料表以垂直方向合併觀測值（列）。
- 水平合併：讓資料表以水平方向合併變數（欄）。

垂直合併

以 `UNION` 垂直合併

```
A SELECT statement
UNION
Another SELECT statement
```

垂直合併簡單來說，就是結合觀測值



In [5]:

```
SELECT director AS my_favorites
  FROM movies
 WHERE director IN ('Christopher Nolan', 'Steven Spielberg')
 UNION
SELECT name
  FROM actors
 WHERE name IN ('Tom Hanks', 'Leonardo DiCaprio');
```

Out[5]:

my_favorites
Christopher Nolan
Leonardo DiCaprio
Steven Spielberg
Tom Hanks

4 rows in set (0.00 sec)

使用 `UNION` 的注意事項

- 垂直合併的欄位數要相同。
- 垂直合併的 SQL 若有使用到 `ORDER BY` 只能放在 `UNION` 之後。
- 垂直合併的重複觀測值會被省略。

垂直合併的欄位數要相同

```sql
SELECT director AS my_favorites
  FROM movies
 WHERE director IN ('Christopher Nolan', 'Steven Spielberg')
 UNION
SELECT name,
       id  -- 垂直合併的欄位數不相同
  FROM actors
 WHERE name IN ('Tom Hanks', 'Leonardo DiCaprio');
```

In [ ]:

垂直合併的 SQL 若有使用到 `ORDER BY` 只能放在 `UNION` 之後

```sql
SELECT director AS my_favorites
  FROM movies
 WHERE director IN ('Christopher Nolan', 'Steven Spielberg')
 ORDER BY my_favorites
 UNION
SELECT name
  FROM actors
 WHERE name IN ('Tom Hanks', 'Leonardo DiCaprio');
```

In [ ]:

垂直合併的重複觀測值會被省略

In [6]:

```
SELECT director AS my_favorites
  FROM movies
 WHERE director IN ('Christopher Nolan', 'Steven Spielberg');
SELECT name
  FROM actors
 WHERE name IN ('Tom Hanks', 'Leonardo DiCaprio')
 ORDER BY my_favorites;
```

Out[6]:

my_favorites
Christopher Nolan
Steven Spielberg
Christopher Nolan
Steven Spielberg
Christopher Nolan
Christopher Nolan
Christopher Nolan
Steven Spielberg
Christopher Nolan
Steven Spielberg
Christopher Nolan
Steven Spielberg
Steven Spielberg

13 rows in set (0.00 sec)

若希望保留重複觀測值，改使用 `UNION ALL`

```
A SELECT statement
UNION ALL
Another SELECT statement
```

In [7]:

```
SELECT director AS my_favorites
  FROM movies
 WHERE director IN ('Christopher Nolan', 'Steven Spielberg')
 UNION ALL
SELECT name
  FROM actors
 WHERE name IN ('Tom Hanks', 'Leonardo DiCaprio')
 ORDER BY my_favorites;
```

Out[7]:

my_favorites
Christopher Nolan
Christopher Nolan
Christopher Nolan
Christopher Nolan
Christopher Nolan
Christopher Nolan
Christopher Nolan
Leonardo DiCaprio
Steven Spielberg
Steven Spielberg
Steven Spielberg
Steven Spielberg
Steven Spielberg
Steven Spielberg
Tom Hanks

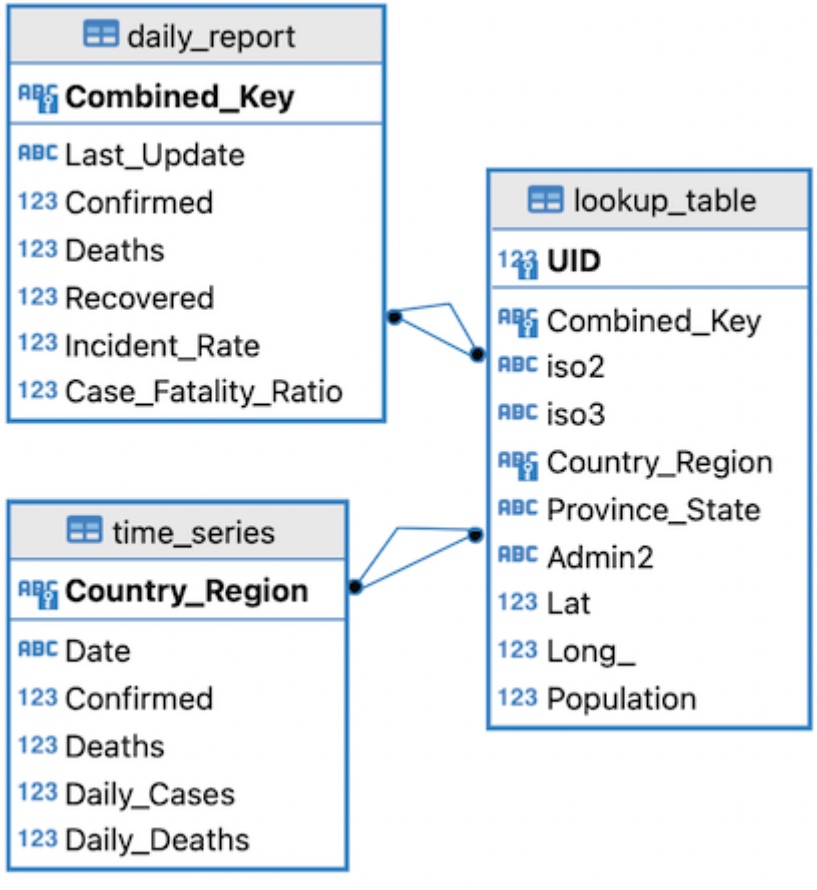
15 rows in set (0.00 sec)

水平合併

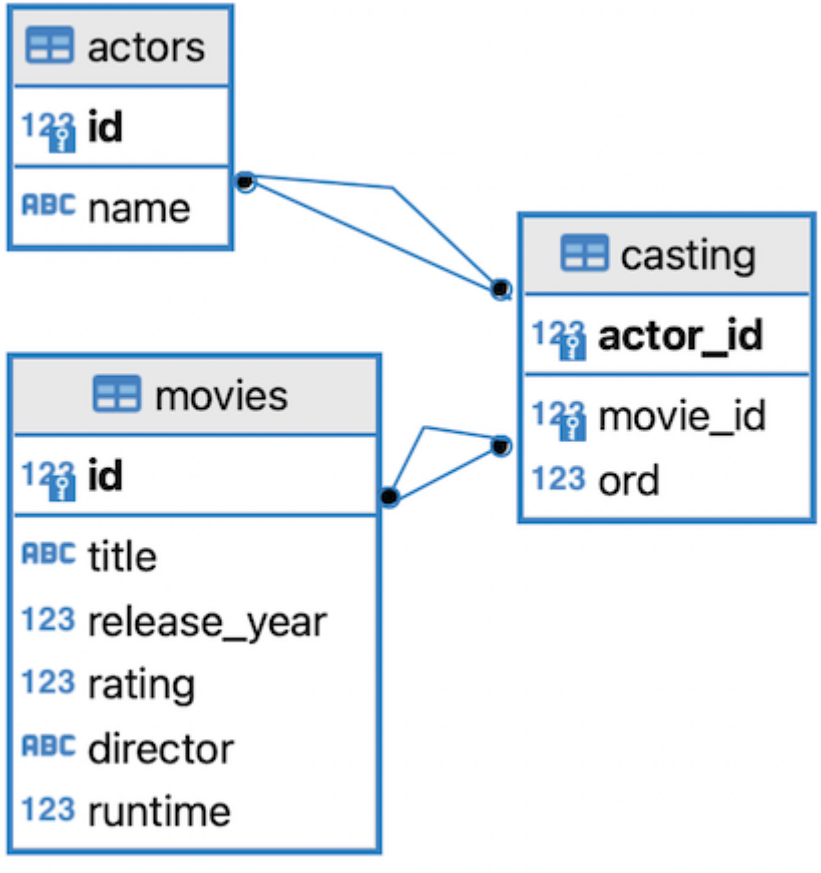
學習資料庫的實體關係圖（ER Diagram, Entity Relationship Diagram）

- 體會關聯式資料庫的核心精神。
- 瞭解資料表之間透過如同「橋樑」般存在的**結合鍵**（Join Key）水平合併。

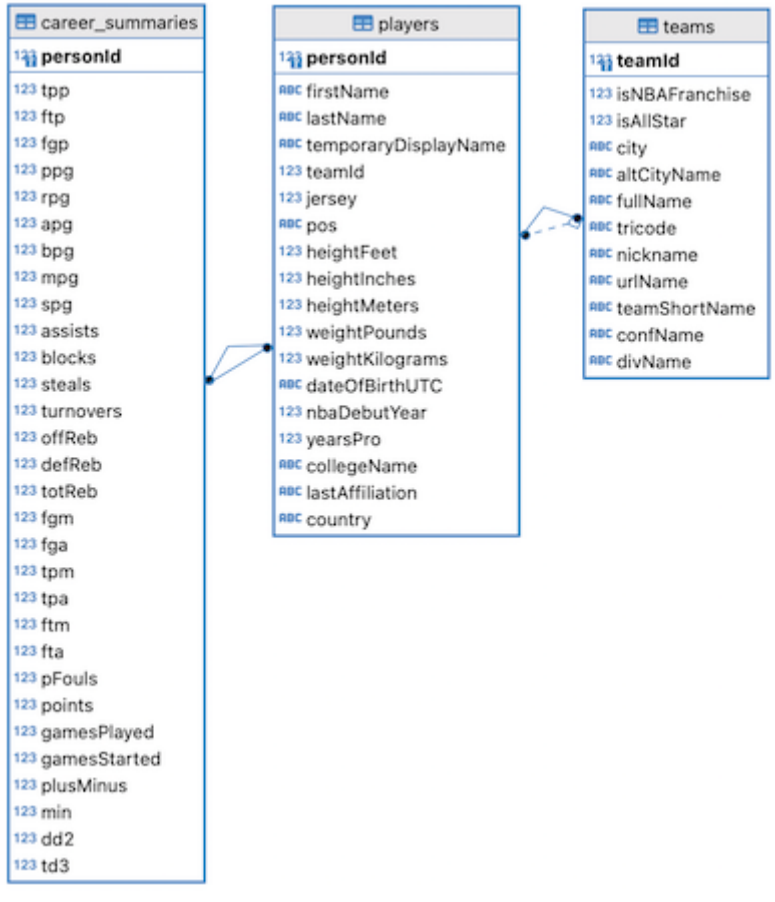
`covid19` 資料庫的實體關係圖



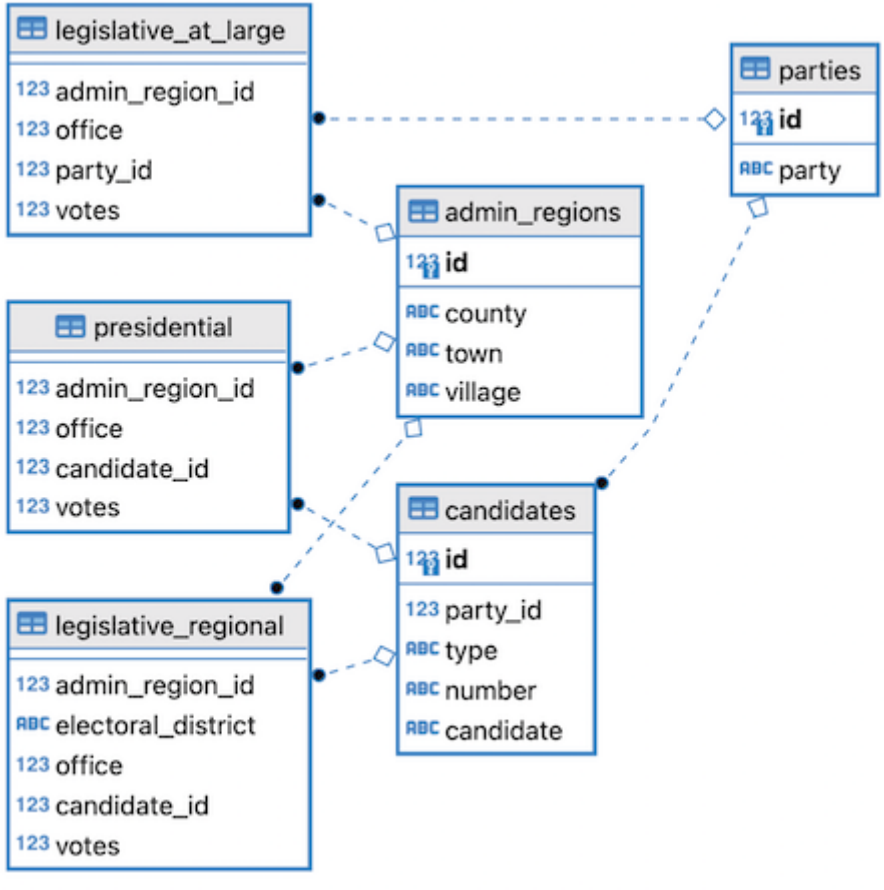
`imdb` 資料庫的實體關係圖



nba 資料庫的實體關係圖



`twElection2020` 資料庫的實體關係圖

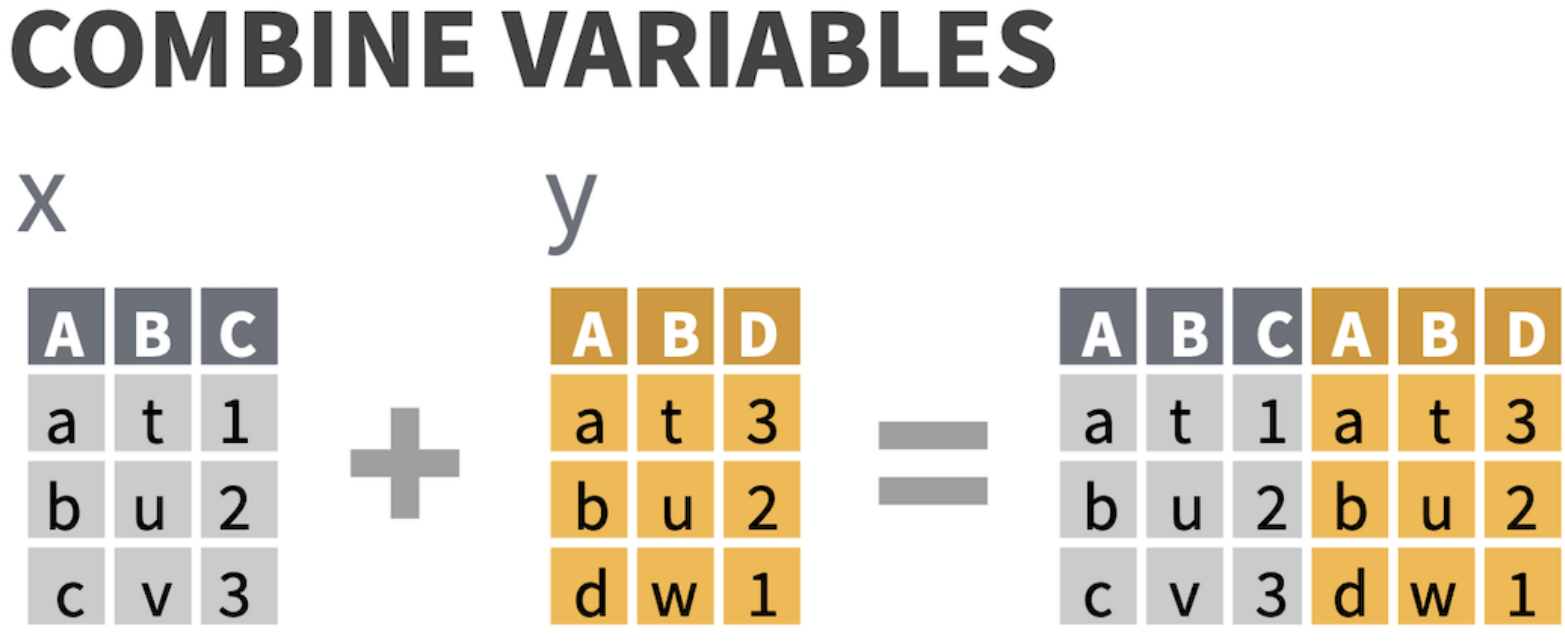


以 `JOIN` 水平合併

因為是「水平」合併，在 `FROM` 後的資料表被稱為「左表格」、`JOIN` 後的資料表被稱為「右表格」。

```sql
SELECT left_table.column_names,
       right_table.column_names
  FROM left_table
  JOIN right_table
    ON left_table.join_key = right_table.join_key;
```

水平合併簡單來說，就是結合變數



以 `JOIN` 水平合併 `movies` 與 `casting`

In [8]:

```sql
SELECT movies.title,
       casting.actor_id
  FROM movies  -- 左表格
  JOIN casting -- 右表格
    ON movies.id = casting.movie_id
 WHERE movies.title = 'The Shawshank Redemption';
```

Out[8]:

title	actor_id
The Shawshank Redemption	2853
The Shawshank Redemption	2097
The Shawshank Redemption	333
The Shawshank Redemption	3029
The Shawshank Redemption	533
The Shawshank Redemption	1042
The Shawshank Redemption	1923
The Shawshank Redemption	1319
The Shawshank Redemption	1370
The Shawshank Redemption	1724
The Shawshank Redemption	2151
The Shawshank Redemption	363
The Shawshank Redemption	659
The Shawshank Redemption	1530
The Shawshank Redemption	1553

15 rows in set (0.00 sec)

以 JOIN 水平合併 movies、casting 與 actors

In [9]:

```
SELECT movies.title,
       casting.ord,
       actors.name
  FROM movies    -- 左表格
  JOIN casting   -- 右一表格
    ON movies.id = casting.movie_id
  JOIN actors    -- 右二表格
    ON casting.actor_id = actors.id
 WHERE movies.title = 'The Shawshank Redemption';
```

Out[9]:

title	ord	name
The Shawshank Redemption	1	Tim Robbins
The Shawshank Redemption	2	Morgan Freeman
The Shawshank Redemption	3	Bob Gunton
The Shawshank Redemption	4	William Sadler
The Shawshank Redemption	5	Clancy Brown
The Shawshank Redemption	6	Gil Bellows
The Shawshank Redemption	7	Mark Rolston
The Shawshank Redemption	8	James Whitmore
The Shawshank Redemption	9	Jeffrey DeMunn
The Shawshank Redemption	10	Larry Brandenburg
The Shawshank Redemption	11	Neil Giuntoli
The Shawshank Redemption	12	Brian Libby
The Shawshank Redemption	13	David Proval
The Shawshank Redemption	14	Joseph Ragno
The Shawshank Redemption	15	Jude Ciccolella

15 rows in set (0.00 sec)

使用 `JOIN` 水平合併資料表時的注意事項

- 養成為水平合併的資料表取別名並在欄位名稱前註明清楚的好習慣。
- 除了實體資料表亦可使用子查詢生成的結果作為合併來源。
- 預設保留左表格與右表格**交集**的觀測值。

養成為水平合併的資料表取別名並在欄位名稱前註明清楚的好習慣

舉例來說，在 `movies` 與 `actors` 兩個資料表中都有 `id` 欄位。

- `movies.id` 指的是電影編號。
- `actors.id` 指的是演員編號。

如果沒有註明 id 來自哪個資料表會產生錯誤

```
SELECT title,
       ord,
       name
  FROM movies
  JOIN casting
    ON id = movie_id
  JOIN actors
    ON actor_id = id
 WHERE title = 'The Shawshank Redemption';
```

In [ ]:

除了實體資料表亦可使用子查詢生成的結果作為合併來源

In [10]:

```sql
SELECT subquery_movies.title,
       casting.ord,
       actors.name
  FROM (SELECT id,
               title,
               director
          FROM movies
         WHERE title IN ('The Shawshank Redemption', 'Forrest Gump')) AS subquery_movies -- 左表格
  JOIN casting  -- 右一表格
    ON subquery_movies.id = casting.movie_id
  JOIN actors   -- 右二表格
    ON casting.actor_id = actors.id
 ORDER BY ord;
```

Out[10]:

title	ord	name
The Shawshank Redemption	1	Tim Robbins
Forrest Gump	1	Tom Hanks
The Shawshank Redemption	2	Morgan Freeman
Forrest Gump	2	Rebecca Williams
The Shawshank Redemption	3	Bob Gunton
Forrest Gump	3	Sally Field
The Shawshank Redemption	4	William Sadler
Forrest Gump	4	Michael Conner Humphreys
The Shawshank Redemption	5	Clancy Brown
Forrest Gump	5	Harold G. Herthum
The Shawshank Redemption	6	Gil Bellows
Forrest Gump	6	George Kelly
The Shawshank Redemption	7	Mark Rolston
Forrest Gump	7	Bob Penny
The Shawshank Redemption	8	James Whitmore
Forrest Gump	8	John Randall
The Shawshank Redemption	9	Jeffrey DeMunn
Forrest Gump	9	Sam Anderson
The Shawshank Redemption	10	Larry Brandenburg

Forrest Gump	10	Margo Moorer
The Shawshank Redemption	11	Neil Giuntoli
Forrest Gump	11	Ione M. Telech
The Shawshank Redemption	12	Brian Libby
Forrest Gump	12	Christine Seabrook
The Shawshank Redemption	13	David Proval
Forrest Gump	13	John Worsham
The Shawshank Redemption	14	Joseph Ragno
Forrest Gump	14	Peter Dobson
The Shawshank Redemption	15	Jude Ciccolella
Forrest Gump	15	Siobhan Fallon Hogan

30 rows in set (0.00 sec)

預設保留左表格與右表格交集的觀測值

In [11]:

```
SELECT two_movies.title,
       two_castings.actor_id
  -- 左表格是刺激1995與阿甘正傳的電影資料
  FROM (SELECT *
          FROM movies
         WHERE title IN ('The Shawshank Redemption', 'Forrest Gump')) AS two_movies
  -- 右表格是刺激1995與黑暗騎士的演員名單資料
  JOIN (SELECT *
          FROM casting
         WHERE movie_id IN (1, 4)) AS two_castings
    ON two_movies.id = two_castings.movie_id
 ORDER BY two_movies.title;
```

Out[11]:

title	actor_id
The Shawshank Redemption	2853
The Shawshank Redemption	2097
The Shawshank Redemption	333
The Shawshank Redemption	3029
The Shawshank Redemption	533
The Shawshank Redemption	1042
The Shawshank Redemption	1923
The Shawshank Redemption	1319
The Shawshank Redemption	1370
The Shawshank Redemption	1724
The Shawshank Redemption	2151
The Shawshank Redemption	363
The Shawshank Redemption	659
The Shawshank Redemption	1530
The Shawshank Redemption	1553

15 rows in set (0.00 sec)

若希望保留以左表格為主的觀測值，改使用 `LEFT JOIN`

```sql
SELECT left_table.column_names,
       right_table.column_names
  FROM table_name AS left_table
  LEFT JOIN table_name AS right_table
    ON left_table.join_key = right_table.join_key;
```

In [12]:

```
SELECT two_movies.title,
       two_castings.actor_id
  -- 左表格是刺激1995與阿甘正傳的電影資料
  FROM (SELECT *
          FROM movies
         WHERE title IN ('The Shawshank Redemption', 'Forrest Gump')) AS two_movies
  -- 右表格是刺激1995與黑暗騎士的名單資料
  LEFT JOIN (SELECT *
               FROM casting
              WHERE movie_id IN (1, 4)) AS two_castings
    ON two_movies.id = two_castings.movie_id
 ORDER BY two_movies.title;
```

Out[12]:

title	actor_id
Forrest Gump	NULL
The Shawshank Redemption	333
The Shawshank Redemption	363
The Shawshank Redemption	533
The Shawshank Redemption	659
The Shawshank Redemption	1042
The Shawshank Redemption	1319
The Shawshank Redemption	1370
The Shawshank Redemption	1530
The Shawshank Redemption	1553
The Shawshank Redemption	1724
The Shawshank Redemption	1923
The Shawshank Redemption	2097
The Shawshank Redemption	2151
The Shawshank Redemption	2853
The Shawshank Redemption	3029

16 rows in set (0.01 sec)

重點統整

- 我們的查詢範圍開始由「單個」資料表擴展至「多個」資料表。
- 資料表能夠用兩個方向關聯：
    - 垂直合併
    - 水平合併

重點統整（續）

- 使用 `UNION` 垂直合併的注意事項
    - 垂直合併的欄位數要相同。
    - 垂直合併的 SQL 若有使用到 `ORDER BY` 只能放在 `UNION` 之後。
    - 垂直合併的重複觀測值會被省略。
    - 若希望保留重複觀測值，改使用 `UNION ALL`。

重點統整（續）

- 使用 `JOIN` 水平合併的注意事項
    - 養成為水平合併的資料表取別名並在欄位名稱前註明清楚的好習慣。
    - 除了實體資料表亦可使用子查詢生成的結果作為合併來源。
    - 預設保留左表格與右表格交集的觀測值。
    - 若希望保留以左表格為主的觀測值，改使用 `LEFT JOIN`。

In [ ]:

```sql
/*截至目前學起來的 SQL 有哪些？
SQL 寫作順序必須遵從標準 SQL 的規定。*/
SELECT column_names      -- 選擇哪些欄位
  FROM left_table        -- 從哪個資料庫的資料表
  JOIN right_table       -- 與哪個資料表水平合併
    ON left_table.join_key = right_table.join_key
 WHERE conditions        -- 篩選哪些觀測值
 GROUP BY column_names   -- 指定依照哪個變數分組
HAVING conditions        -- 篩選哪些分組聚合的結果
 UNION SELECT statement  -- 與哪段 SQL 垂直合併
 ORDER BY column_names   -- 指定依照哪個變數排序
 LIMIT m;                -- 查詢結果顯示前 m 列就好
```