

DBMS_Project_Document

F74064012_黃冠淳

1. Project 簡介&目標

由於現在 NBA 季後賽正如火如荼地進行，NBA 球迷們都十分關注各個賽況，而此專案的主要目的就是幫助球迷們能快速搜尋球隊資訊，包含球員數據、比賽紀錄等，讓球迷們能預測及討論哪支球隊能夠贏下東西區冠軍，甚至是總冠軍，增加觀賽趣味。

2. 系統架構與環境 & 介面截圖與使用說明

2.1 系統架構與環境

2.1.1 系統架構：

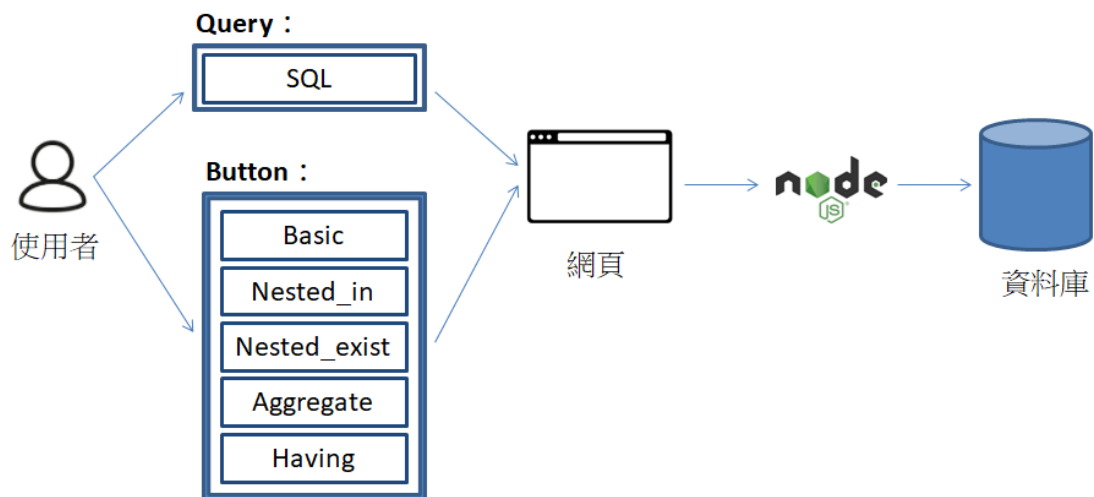


Figure 2.1.1 系統架構圖

2.1.2 環境：

- 作業系統：Linux
- 資料庫：MySQL
- 後端框架：Nodejs
- 使用語言：Javascript,css,html
- 執行方式：node ser.js

2.2 介面截圖與使用說明

2.2.1: Query

SQLbasicNested_INNested_ExistsAggregatehaving

輸入SQL指令:

SELECT player_name FROM player
WHERE PTS>=30;

1. 輸入 SQL 指令

執行結果:

確認

2. 按下確認鍵

player_name
Bradley Beal

3. 顯示執行結果

Figure 2. 2. 1 Query

2. 2. 2: Button

- Basic:

SELECT-FROM-WHERE , DELETE , INSERT , UPDATE

- 範例 SQL 指令:

```
SELECT player_name,PTS,REB,AST FROM player
WHERE PTS>=30;
```

SQL

basic

Nested_IN

Nested_Exists

Aggregate

having

選擇修改方式:

查詢

 → 1. 選擇查詢/刪除/新增/更改

選擇修改資料表:

player

 → 2. 選擇資料表

輸入要查詢的條件:

欄位	值
Pnumber	<input type="text"/>
player_name	<input type="text"/>
PTS	<div>>=30</div> → 3. 輸入條件
REB	<input type="text"/>
AST	<input type="text"/>
Tno	<input type="text"/>
HCno	<input type="text"/>

確認

 → 4. 按下確認鍵

輸入要查詢的欄位:

player_name

PTS

REB

AST

 → 5. 輸入欄位

確認查詢

 → 6. 確認查詢

查詢結果:

player_name	PTS	REB	AST
Bradley Beal	31.3	4.7	4.4

 → 7. 顯示結果

Figure 2. 2. 2 SELECT-FROM-WHERE

2. 2. 3: Button

- Nested Queries :

IN/NOT IN

- 範例 SQL 指令:

SELECT player_name FROM player

WHERE Tno

in (SELECT Tnumber FROM team WHERE team_group="太平洋組");

SQL basic **Nested_IN** Nested_Exists Aggregate having

選擇查詢資料表:

player

 → 1. 選擇資料表

輸入要查詢的欄位:

player_name

 → 2. 輸入查詢欄位

WHERE

Tno

 → 3. 選擇條件欄位

IN

 → 4. 選擇 IN/NOT IN

team

Tnumber

 → 5. 選擇子查詢資料表&欄位

條件:

欄位	值
Tnumber	
team_name	
team_alliance	
team_champion	
Cno	
team_group	= "太平洋組" → 6. 輸入條件

確認查詢

 → 7. 確認查詢

查詢結果:

player_name
Lebron James
Chris Paul
Devin Booker
Kawhi Leonard
Paul George

 → 8. 顯示結果

Figure 2. 2. 3 IN

2. 2. 4 : Button

- **Nested Queries :**

EXISTS/NOT EXISTS

- **範例 SQL 指令:**

```
SELECT player_name,PTS FROM player
```

```
WHERE EXISTS
```

```
(SELECT * FROM team WHERE Tnumber=Tno AND  
team_alliance="東區");
```

SQL basic Nested_IN Nested_Exists Aggregate having

選擇查詢資料表:

player

1. 選擇資料表

輸入要查詢的欄位:

player_name
PTS

2. 輸入欄位

WHERE EXISTS

3. 選擇 EXISTS/NOT EXISTS

team

4. 選擇子查詢資料表

條件:

欄位	值
Tnumber	=Tno
team_name	
team_alliance	= "東區"
team_champion	
Cno	
team_group	

5. 輸入條件

確認查詢

6. 確認查詢

查詢結果:

player_name	PTS
Derrick Rose	14.9
James Harden	20.5
Joel Embiid	28.5
Bradley Beal	31.3
Kevin Durant	26.9
Kyrie Irving	26.9
Jayson Tatum	26.4
Trae Young	25.3

7. 顯示結果

Figure 2. 2. 4 EXISTS

2. 2. 5 : Button

- **Aggregate functions :**

COUNT, SUM, MAX, MIN, AVG

- **範例 SQL 指令:**

```
SELECT COUNT(game_date) FROM game_record WHERE  
game_date='2021-05-24';
```

SQL basic Nested_IN Nested_Exists **Aggregate** having

選擇查詢資料表:

game_record ▼ game_date ▼

→ 1. 選擇資料表&欄位

選擇Aggregate functions:

COUNT ▼

→ 2. 選擇 Aggregate functions

條件:

欄位	值
GRnumber	<input type="text"/>
win_Tnumber	<input type="text"/>
lose_Tnumber	<input type="text"/>
Cno	<input type="text"/>
game_date	<input type="text" value="'2021-05-24'"/>
score	<input type="text"/>

→ 3. 輸入條件

確認查詢 → 4. 確認查詢

查詢結果:

COUNT(game_date)

2

→ 5. 顯示結果

Figure 2. 2. 5 COUNT

2.2.6 : Button

- **Aggregate functions :**

HAVING

- **範例 SQL 指令:**

```
SELECT team_group, SUM(team_champion) FROM team
GROUP BY team_group
HAVING SUM(team_champion)>5;
```

SQL basic Nested_IN Nested_Exists Aggregate **having**

選擇查詢資料表:

 → 1. 選擇資料表

選擇Aggregate functions:

 → 2. 選擇 Aggregate functions

選擇Aggregate 欄位:

 → 3. 選擇 Aggregate 欄位

WHERE

HAVING

Figure 2.2.6 having

3. 資料庫設計

3.1 ER diagram & 說明

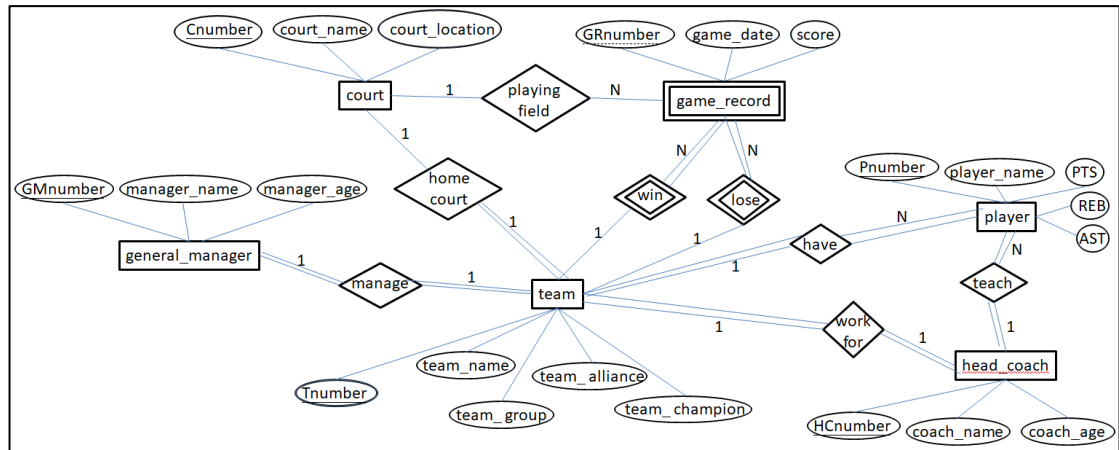


Figure 3. 1 ER diagram

3.1.1 Entity:

- team(球隊)
- player(球員)
- head_coach(總教練)
- game_record(比賽紀錄) – Weak Entity
- court(主場球館)
- general_manager(總經理)

3.1.2 attribute:

- team
 - Tnumber (key attribute)
 - team_name(球隊名稱)
 - team_group(球隊分組)
 - team_alliance(球隊所屬聯盟)
 - team_champion(球隊總冠軍數)
- player
 - Pnumber(key attribute)
 - player_name(球員名字)
 - PTS(球員賽季平均得分)
 - AST(球員賽季平均助攻)
 - REB(球員賽季平均籃板)

- **head_coach**
 HCnumber(key attribute)
 coach_name(總教練名字)
 coach_age(總教練年齡)
- **game_record**
 GRnumber(partial key)
 game_date(比賽日期)
 score(比賽比分)
- **court**
 Cnumber(key attribute)
 court_name(主場館名稱)
 court_location(主場館所在地)
- **general_manager**
 GRnumber(key attribute)
 manager_name(總經理名字)
 manager_age(總經理年齡)

3.1.3 relationship:

- **manage:**
 每個總經理都要(只能)管理一個球隊，且每個球隊都要(只能)被一個總經理管理
- **home_court:**
 每個球隊都要有(只能有)一個主場球館，但球館不一定有隸屬球隊
- **win:**
 有些球隊有贏的紀錄，有些球隊沒有；一個球隊可能有 n 筆贏的紀錄，一筆贏的紀錄只能隸屬一個球隊
- **lose:**
 有些球隊有輸的紀錄，有些球隊沒有；一個球隊可能有 n 筆輸的紀錄，一筆輸的紀錄只能隸屬一個球隊
- **playing field:**
 有些球館有比賽紀錄，有些球館沒有；一個球館可能有 n 筆比賽紀錄，一筆比賽紀錄只能隸屬一個球館
- **have:**
 一個球隊可以有(一定要有)多個球員，每個球員一定要隸屬一個球隊
- **teach:**
 一個總教練可以教(一定要教)多個球員，每個球員一定要被一個總教練教

- work for:

每個總教練都要(只能)為一個球隊工作，且每個球隊都(只能)有一個總教練

3. 2 Relation Schema & 說明

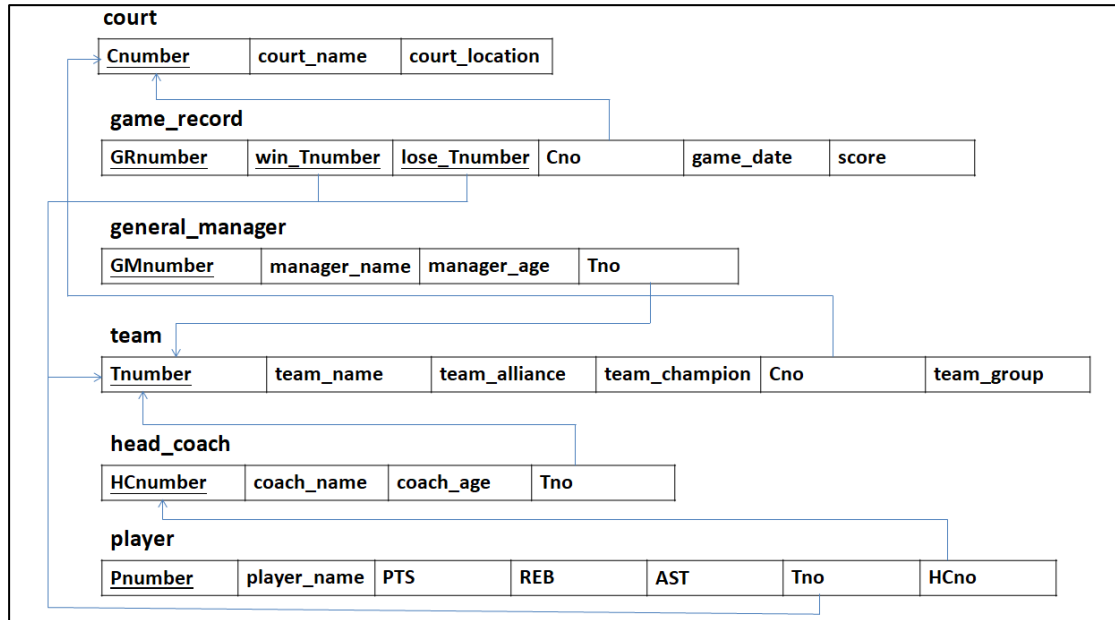


Figure 3. 2 Relation Schema

3. 2. 1 game_record

- game_record(dependent)要把 team(owner)的 key(Tnumber)跟自己的 key(GRnumber)合在一起成為合成的 key
- court 跟 game_record 是 1:N，要把 1 那邊的 key 加到 N 那邊的 table，FOREIGN KEY (Cno) REFERENCES court (Cnumber)

3. 2. 2 general_manager

- general_manager 跟 team 是 1:1，哪一個 key 放到哪一個 table 都可以，FOREIGN KEY (Tno)REFERENCES team (Tnumber)

3. 2. 3 team

- team 跟 court 是 1:1，哪一個 key 放到哪一個 table 都可以，但此例(court 放 team 比較好，因 total participation)
FOREIGN KEY (Cno) REFERENCES court (Cnumber)

3. 2. 4 head_coach

- head_coach 跟 team 是 1:1，哪一個 key 放到哪一個 table 都可以，FOREIGN KEY (Tno) REFERENCES team (Tnumber)

3.2.5 player

- team 跟 palyer 是 1:N，要把 1 那邊的 key 加到 N 那邊的 table，
FOREIGN KEY (Tno) REFERENCES team (Tnumber)
- head_coach 跟 palyer 是 1:N，要把 1 那邊的 key 加到 N 那邊的
table，FOREIGN KEY (HCno) REFERENCES head_coach(HCnumber)