

---

練習

# 網路應用程式與資料庫的整合

---

鄭安翔

ansel\_cheng@hotmail.com

# Lab 1-1 Soccer 專案整合資料庫

## 1. leagueDB資料庫建立及設定

- ❑ 下載安裝 JavaDB Derby
- ❑ Derby建立leagueDB資料庫
  - User Name : SPUB
  - Password: SPUB
- ❑ 執行schema.sql
  - 建立League, ObjectIDs, Player,Registration四個資料表
- ❑ 執行init\_data.sql
  - 新增2筆資料至ObjectIDs及7筆資料至League
- ❑ 驗證資料

# 下載安裝 JavaDB Derby

[https://db.apache.org/derby/derby\\_downloads.html](https://db.apache.org/derby/derby_downloads.html)

The screenshot shows the Apache Derby Downloads page in a web browser. The page title is "Apache Derby: Downloads" and the URL is "db.apache.org/derby/derby\_downloads.html". The page features a navigation bar with links to Home, Quick Start, Download, Community, Documentation, and Resources. The "Download" section is active, showing a list of download links for various Java versions: For Java 9 and Higher, For Java 8 and Higher, For Java 6 and Higher, For Java 1.4 and Higher, For Java 1.3 and Higher, Deprecated Releases, and Change History. Below this, there are three sections: "For Java 9 and Higher" with a link to 10.15.1.3 (March 5, 2019 / SVN 1853019), "For Java 8 and Higher" with links to 10.14.2.0 (May 3, 2018 / SVN 1828579) and 10.13.1.1 (October 25, 2016 / SVN 1766613), and "For Java 6 and Higher".

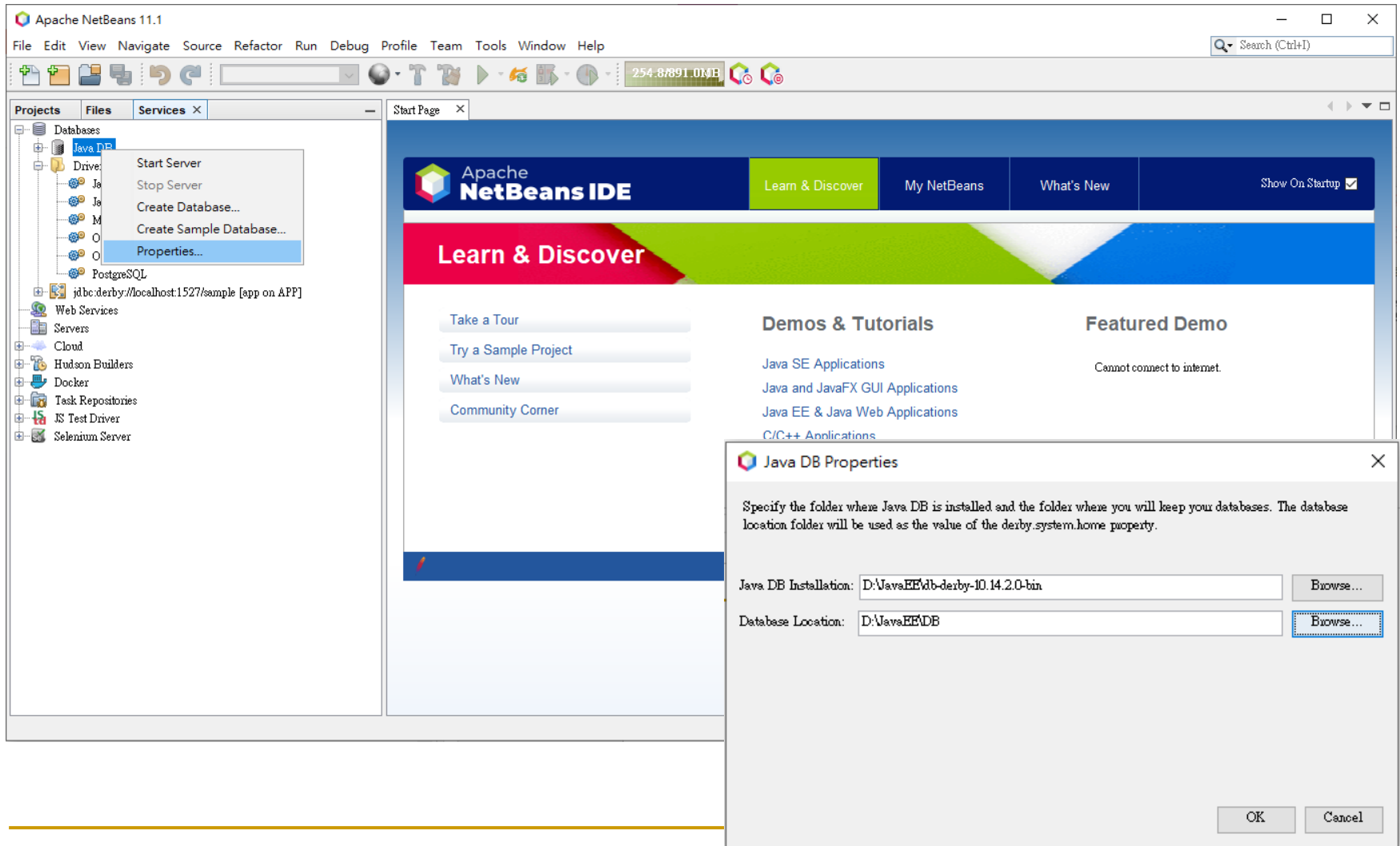
Overlaid on the right side of the screenshot are two file explorer windows. The top window, titled "ClassSetup", shows a table of files:

名稱	修改日期	類型	大小
db-derby-10.14.2.0-bin.zip	2019/11/7 下午 09:...	壓縮的 (zipped) ...	20,664 KB

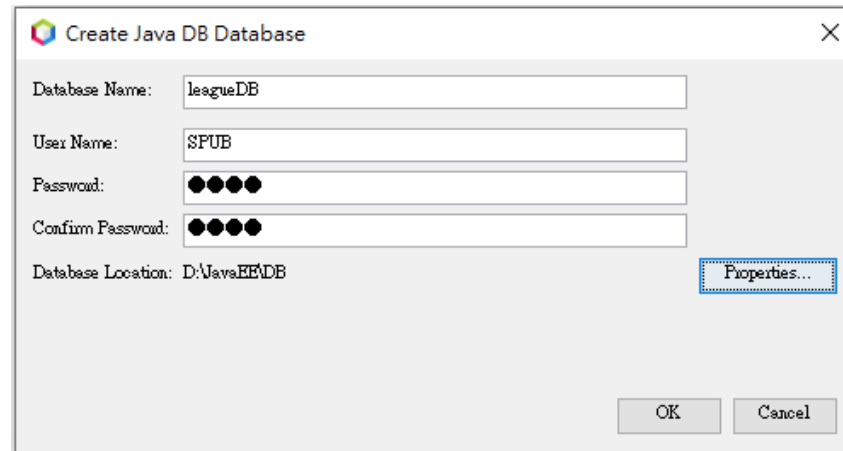
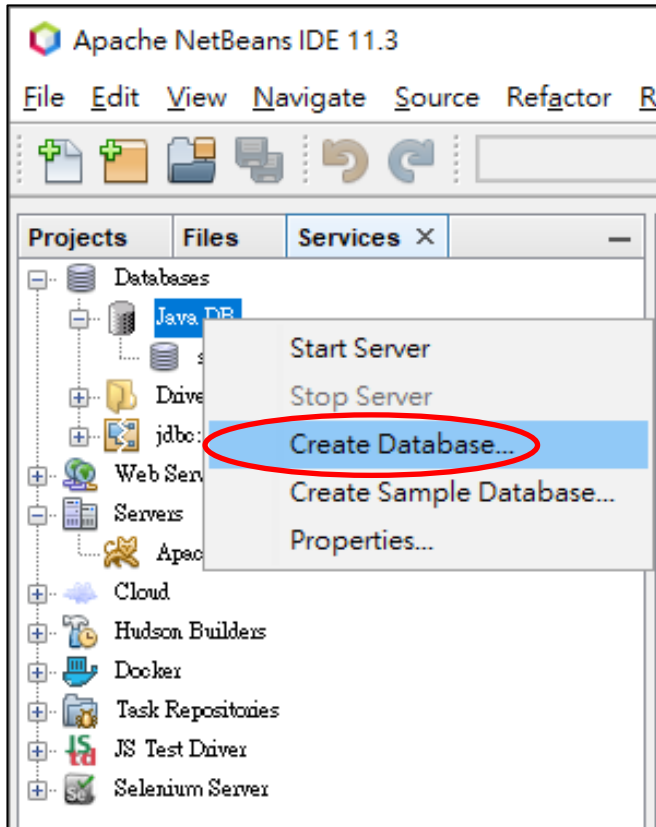
The bottom window, titled "db-derby-10.14.2.0-bin", shows the contents of the extracted zip file:

名稱	修改日期	類型	大小
bin	2019/12/11 下午 10:48	檔案資料夾	
demo	2019/12/11 下午 10:48	檔案資料夾	
docs	2019/12/11 下午 10:49	檔案資料夾	
javadoc	2019/12/11 下午 10:49	檔案資料夾	
lib	2019/12/11 下午 10:49	檔案資料夾	
test	2019/12/11 下午 10:49	檔案資料夾	
index.html	2018/3/10 上午 08:31	Chrome HT...	5 KB
KEYS	2018/4/6 下午 06:14	檔案	47 KB
LICENSE	2018/4/6 下午 06:14	檔案	12 KB
NOTICE	2018/4/6 下午 06:14	檔案	13 KB
RELEASE-NOTES.html	2018/4/6 下午 06:14	Chrome HT...	7 KB

# 連接JavaDB Derby 資料庫



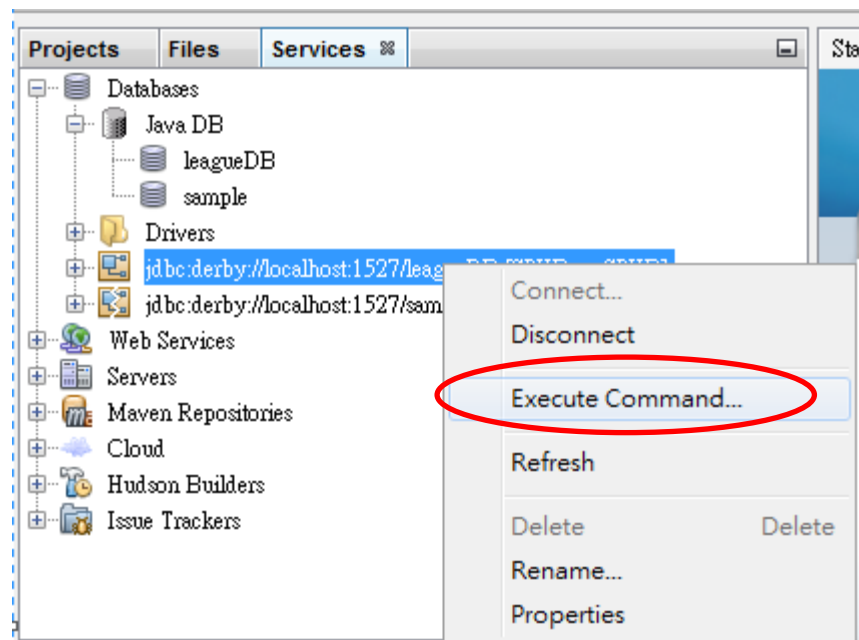
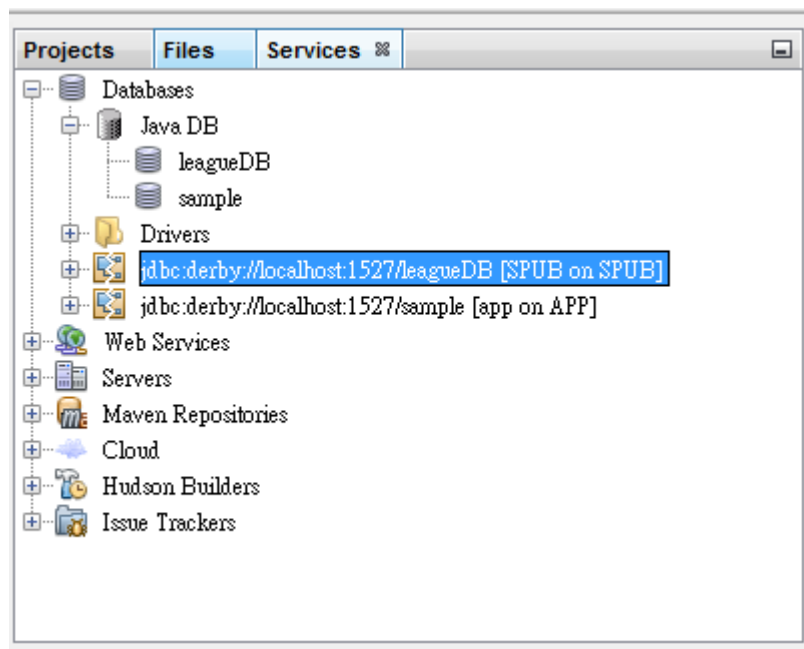
# 建立leagueDB資料庫



User Name : SPUB

Password: SPUB

# 建立資料表



# 建立leagueDB Schema

- 複製schema.sql
  - 建立League, ObjectIDs, Player,Registration四個資料表

The screenshot displays a database IDE interface. On the left, the 'SQL Command 1' window shows a script to drop and create tables. The script includes comments and SQL syntax for creating the 'ObjectIDs' table with a primary key. On the right, the 'Projects' pane shows a hierarchical view of the database schema. The 'leagueDB' database is selected, and its 'Tables' folder is expanded, showing four tables: LEAGUE, OBJECTIDS, PLAYER, and REGISTRATION, which are highlighted with a red rectangle. The 'Drivers' pane also shows the selected connection string: 'jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]'.

```
SQL Command 1
Source History Connection: jdbc:derby://localhost:1527/leagueDB [SPUB o...
6 DROP TABLE Registration;
7 DROP TABLE Player;
8 DROP TABLE League;
9 DROP TABLE ObjectIDs;
10
11
12 --
13 -- This table represents the "next" object_ID for a given ta
14 --
15 CREATE TABLE ObjectIDs (
16 -- PRIMARY KEY --
17 table_name VARCHAR(30) PRIMARY KEY,
18 -- DATA FIELDS --
19 ID_number INTEGER NOT NULL
```

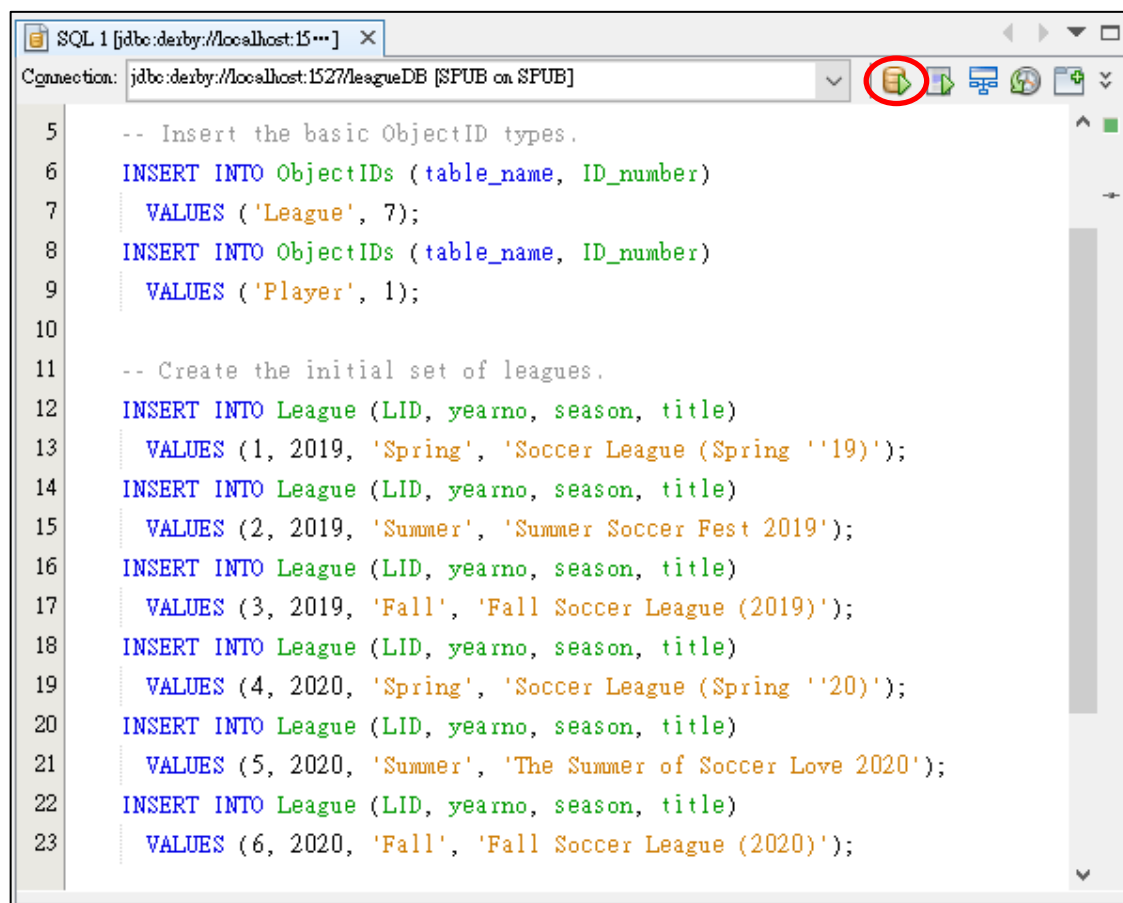
Projects Files Services

- Databases
  - Java DB
    - leagueDB
    - sample
- Drivers
  - jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]**
- APP
- NULLID
- SPUB
  - Tables
    - LEAGUE**
    - OBJECTIDS**
    - PLAYER**
    - REGISTRATION**
  - Views
  - Procedures

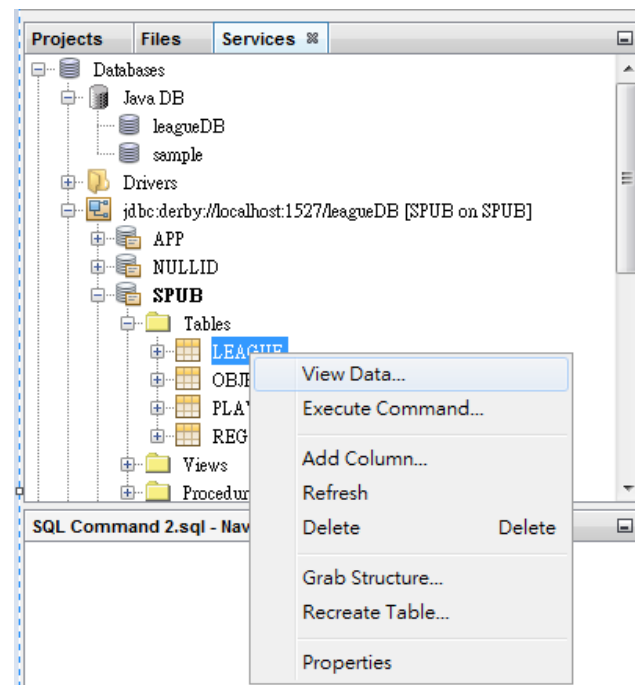
# 建立leagueDB 初始資料

## ■ 複製init\_data.sql

- 建立新增2筆資料至ObjectIDs及6筆資料至League



```
SQL 1 [jdbc:derby://localhost:15...]  
Connection: jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]  
  
5  -- Insert the basic ObjectID types.  
6  INSERT INTO ObjectIDs (table_name, ID_number)  
7     VALUES ('League', 7);  
8  INSERT INTO ObjectIDs (table_name, ID_number)  
9     VALUES ('Player', 1);  
10  
11  -- Create the initial set of leagues.  
12  INSERT INTO League (LID, yearno, season, title)  
13     VALUES (1, 2019, 'Spring', 'Soccer League (Spring '19)');  
14  INSERT INTO League (LID, yearno, season, title)  
15     VALUES (2, 2019, 'Summer', 'Summer Soccer Fest 2019');  
16  INSERT INTO League (LID, yearno, season, title)  
17     VALUES (3, 2019, 'Fall', 'Fall Soccer League (2019)');  
18  INSERT INTO League (LID, yearno, season, title)  
19     VALUES (4, 2020, 'Spring', 'Soccer League (Spring '20)');  
20  INSERT INTO League (LID, yearno, season, title)  
21     VALUES (5, 2020, 'Summer', 'The Summer of Soccer Love 2020');  
22  INSERT INTO League (LID, yearno, season, title)  
23     VALUES (6, 2020, 'Fall', 'Fall Soccer League (2020)');
```





# 檢視資料

SQL 1 [jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]]

SQL 2 [jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]]

Connection: jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]

```

1 SELECT * FROM SPUB.LEAGUE FETCH FIRST 100 ROWS ONLY;
2

```

SELECT \* FROM SPUB.LEAGUE...

Max. rows: 100
Fetched Rows: 6
Matching Rows:

#	LID	YEARNO	SEASON	TITLE
1	1	2019 Spring	Soccer League (Spring '19)	
2	2	2019 Summer	Summer Soccer Fest 2019	
3	3	2019 Fall	Fall Soccer League (2019)	
4	4	2020 Spring	Soccer League (Spring '20)	
5	5	2020 Summer	The Summer of Soccer Love 2020	
6	6	2020 Fall	Fall Soccer League (2020)	

SQL 2 [jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]]

Connection: jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]

```
1 SELECT * FROM SPUB.OBJECTIDS FETCH FIRST 100 ROWS ONLY;
```

SELECT \* FROM SPUB.OBJECT...

Max. rows: 100 | Fetched Rows: 2 | Matching Rows:

#	TABLE_NAME	ID_NUMBER
1	League	7
2	Player	1

SQL 3 [jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]]

Connection: jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]

```
1 SELECT * FROM SPUB.PLAYER FETCH FIRST 100 ROWS ONLY;
```

SELECT \* FROM SPUB.PLAYER...

Max. rows: 100 | Fetched Rows: 0 | Matching Rows:

#	PID	NAME	ADDRESS
---	-----	------	---------

SQL 4 [jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]]

Connection: jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]

```
1 SELECT * FROM SPUB.REGISTRATION FETCH FIRST 100 ROWS ONLY;
```

SELECT \* FROM SPUB.REGIST...

Max. rows: 100 | Fetched Rows: 0 | Matching Rows:

#	LID	PID
---	-----	-----

# Lab 1-2 Soccer專案整合資料庫

1. 資料庫DataSource設定
  - JDBC Driver
  - Context.xml / web.xml
2. 載入並檢視 Soccer專案
  - DAO類別
    - LeagueDAO / ObjectIdDAO
  - Service類別
    - LeagueService
  - 領域物件 Domain Object
    - League
3. 測試執行Soccer專案

# JDBC Driver

Soccer - Apache NetBeans IDE 11.3

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+D)

Projects Files Services

Soccer

- Web Pages
  - META-INF
    - context.xml
  - WEB-INF
    - web.xml
  - admin
    - add\_league.jsp
    - emcx.jsp
    - success.jsp
    - enter\_player.jsp
    - index.html
    - list\_leagues.jsp
    - select\_league.jsp
    - thank\_you.jsp
- Source Packages
  - controller
    - AddLeague.java
    - EnterPlayer.java
    - ListLeagues.java
    - SelectLeague.java
  - model
    - League.java
    - LeagueDAO.java
    - LeagueService.java
    - ObjectidDAO.java
    - Player.java
    - RegistrationService.java
- Libraries
  - derbyLocale\_zh\_TW.jar
- Configuration Files

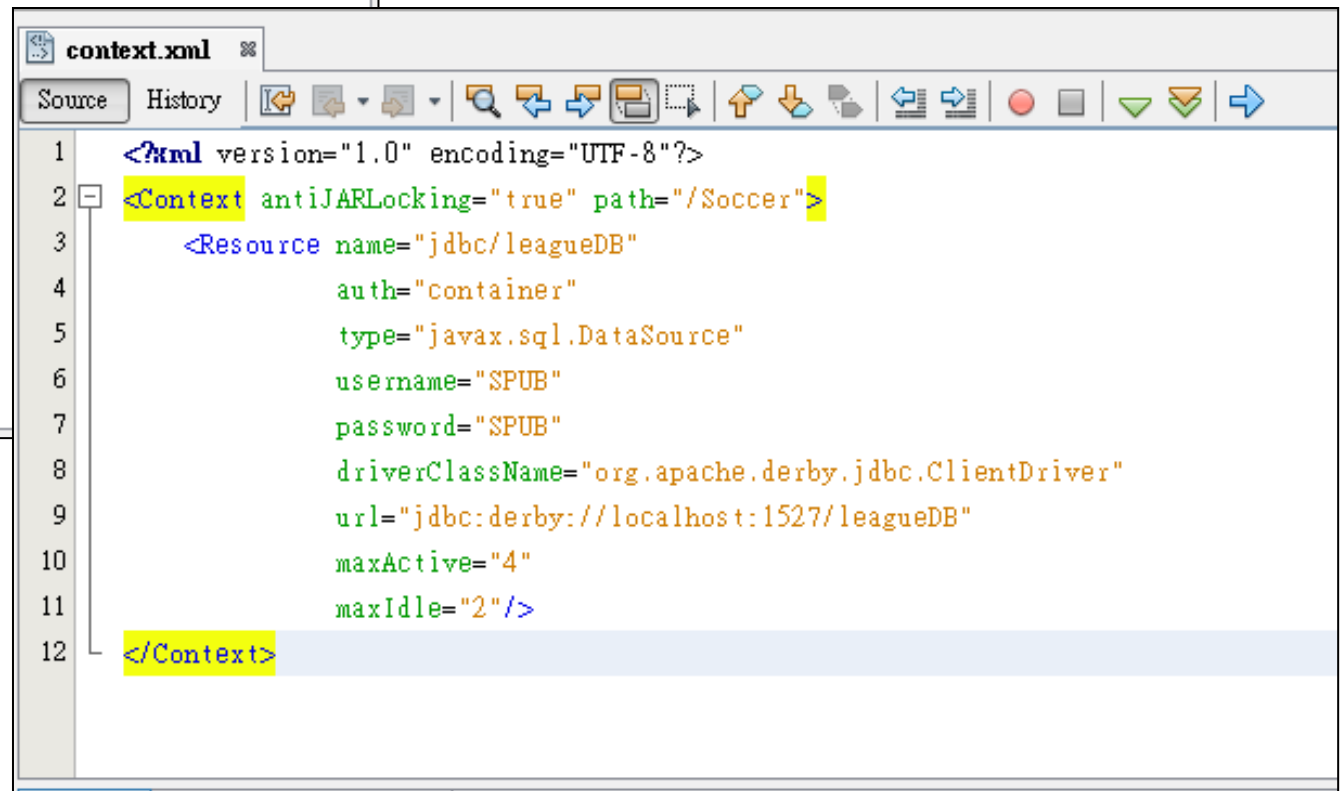
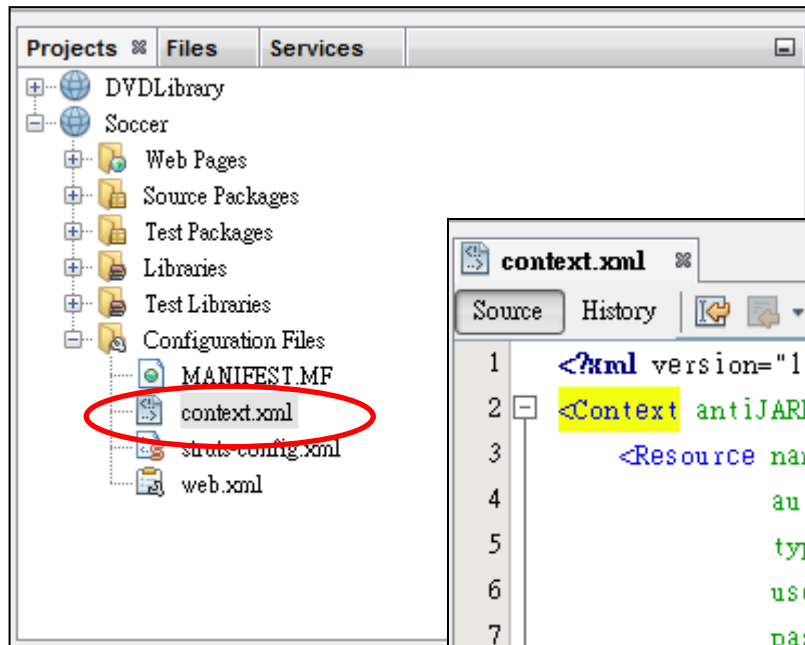
context.xml

```
<Context path="/Soccer">
  <Resource name="jdbc/leagueDB"
    auth="Container"
    type="javax.sql.DataSource"
    username="SPUB"
    password="SPUB"
    driverClassName="org.apache.derby.jdbc.ClientDriver"
    url="jdbc:derby://localhost:1527/leagueDB"
    maxActive="4"
    maxIdle="2"/>
</Context>
```

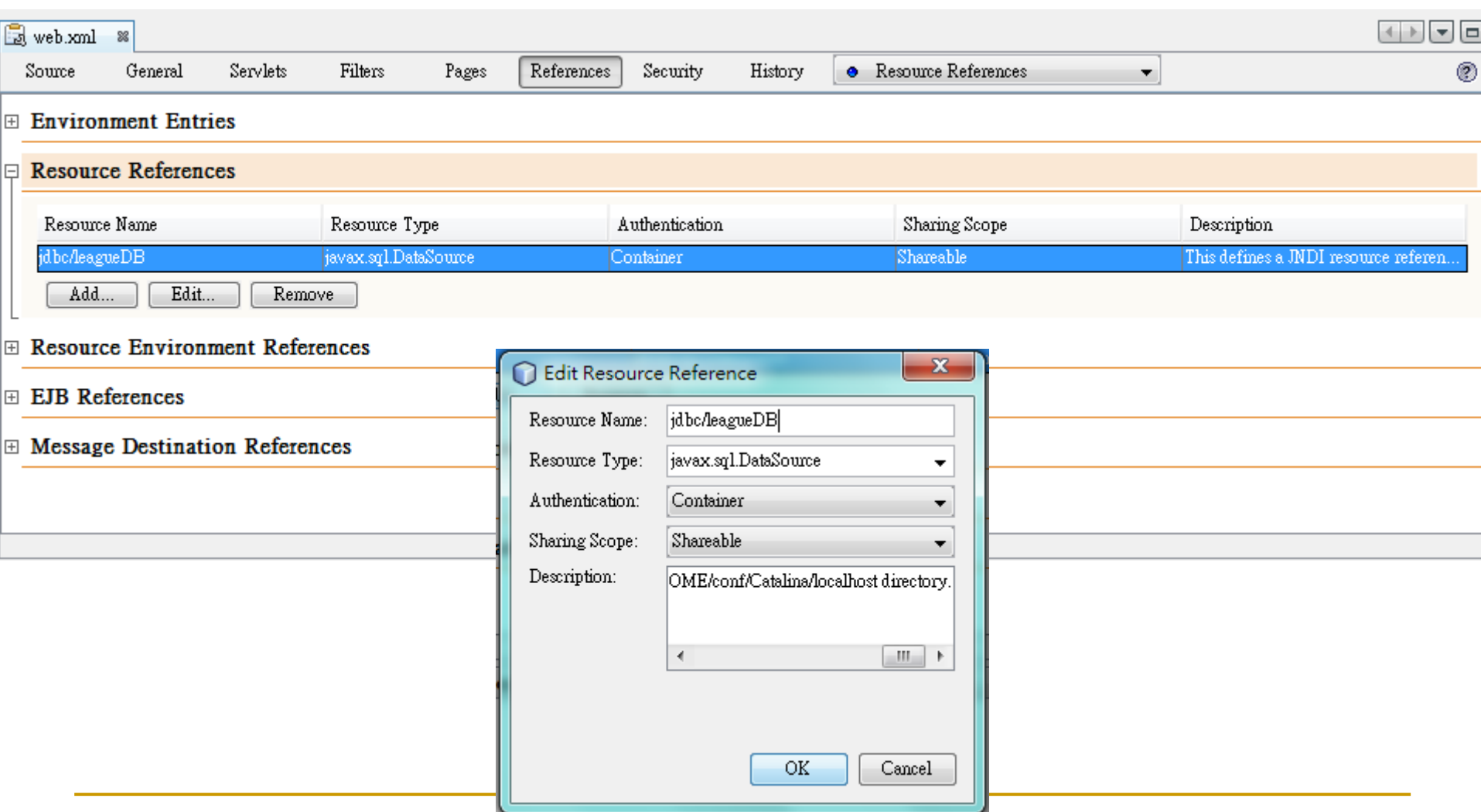
web.xml

```
<web-app version="3.0" xmlns="http://java.sun.com/xml/ns/javaee" xmlns:xsi="
  <context-param>
    <param-name>season-list</param-name>
    <param-value>Spring, Summer, Autumn, Winter</param-value>
  </context-param>
  <session-config>
    <session-timeout>
      30
    </session-timeout>
  </session-config>
  <resource-ref>
    <res-ref-name>jdbc/leagueDB</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
    <res-auth>Container</res-auth>
    <res-sharing-scope>Shareable</res-sharing-scope>
  </resource-ref>
</web-app>
```

# Context.xml



# web.xml設定



Duke's Soccer League: Home

localhost:8080/Socc...

## Duke's Soccer League: Home

This is the Home page for Duke's Soccer League.

### Players

- [List all leagues](#)
- [Register for a league](#)

### League Administrator

- [Add a new league](#)

SQL 2 [jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]]

Connection: jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]

```
1 SELECT * FROM SPUB.OBJECTIDS FETCH FIRST 100 ROWS ONLY;
```

SELECT \* FROM SPUB.OBJECT... x

Max. rows: 100 | Fetched Rows: 2 | Matching Rows:

#	TABLE_NAME	ID_NUMBER
1	League	7
2	Player	1

Duke's Soccer League: List Leag

localhost:8080/Socc...

## Duke's Soccer League: List Leagues

The set of soccer leagues:

- Soccer League (Spring '19)
- Summer Soccer Fest 2019
- Fall Soccer League (2019)
- Soccer League (Spring '20)
- The Summer of Soccer Love 2020
- Fall Soccer League (2020)

End of list.

SQL 1 [jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]]

Connection: jdbc:derby://localhost:1527/leagueDB [SPUB on SPUB]

```
1 SELECT * FROM SPUB.LEAGUE FETCH FIRST 100 ROWS ONLY;
```

SELECT \* FROM SPUB.LEAGUE... x

Max. rows: 100 | Fetched Rows: 6 | Matching Rows:

#	LID	YEARNO	SEASON	TITLE
1	1	2019	Spring	Soccer League (Spring '19)
2	2	2019	Summer	Summer Soccer Fest 2019
3	3	2019	Fall	Fall Soccer League (2019)
4	4	2020	Spring	Soccer League (Spring '20)
5	5	2020	Summer	The Summer of Soccer Love 2020
6	6	2020	Fall	Fall Soccer League (2020)

Duke's Soccer League: Add a New League

This form allows you to create a new soccer league.

Year:

Season:

Title:

Duke's Soccer League: Add League Success

Your request to add the *2020 Winter Game* league was successful.

Duke's Soccer League: List Leagues

The set of soccer leagues:

- Soccer League (Spring '19)
- Summer Soccer Fest 2019
- Fall Soccer League (2019)
- Soccer League (Spring '20)
- The Summer of Soccer Love 2020
- Fall Soccer League (2020)
- 2020 Winter Game

End of list.

SELECT \* FROM SPUB.OBJECT...

Max. rows: 100 | Fetched Rows: 2 |

#	TABLE_NAME	ID_NUMBER
1	League	8
2	Player	1

SELECT \* FROM SPUB.LEAGUE...

Max. rows: 100 | Fetched Rows: 7 |

#	LID	YEARNO	SEASON	TITLE
1	1	2019	Spring	Soccer League (Spring '19)
2	2	2019	Summer	Summer Soccer Fest 2019
3	3	2019	Fall	Fall Soccer League (2019)
4	4	2020	Spring	Soccer League (Spring '20)
5	5	2020	Summer	The Summer of Soccer Love 2020
6	6	2020	Fall	Fall Soccer League (2020)
7	7	2020	Winter	2020 Winter Game

# Lab 2 球員註冊功能整合資料庫

## 1. 球員註冊功能整合資料庫設計

- DAO類別

- PlayerDAO / RegistrationDAO

- Service類別

- RegistrationService / LeagueService

- Exception類別

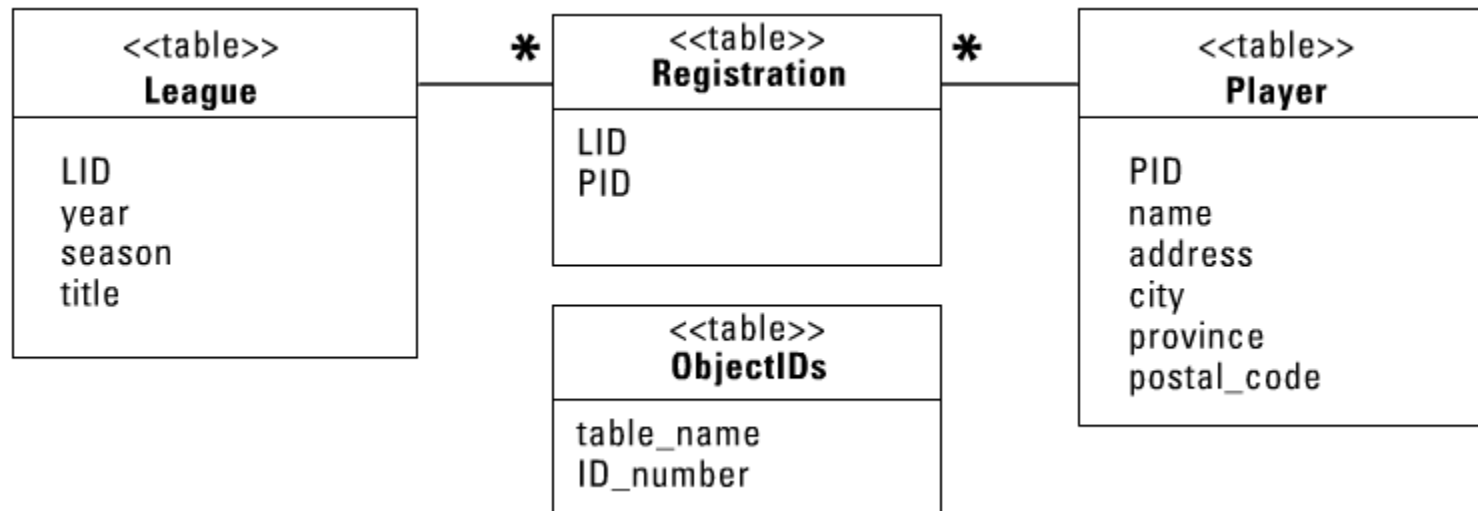
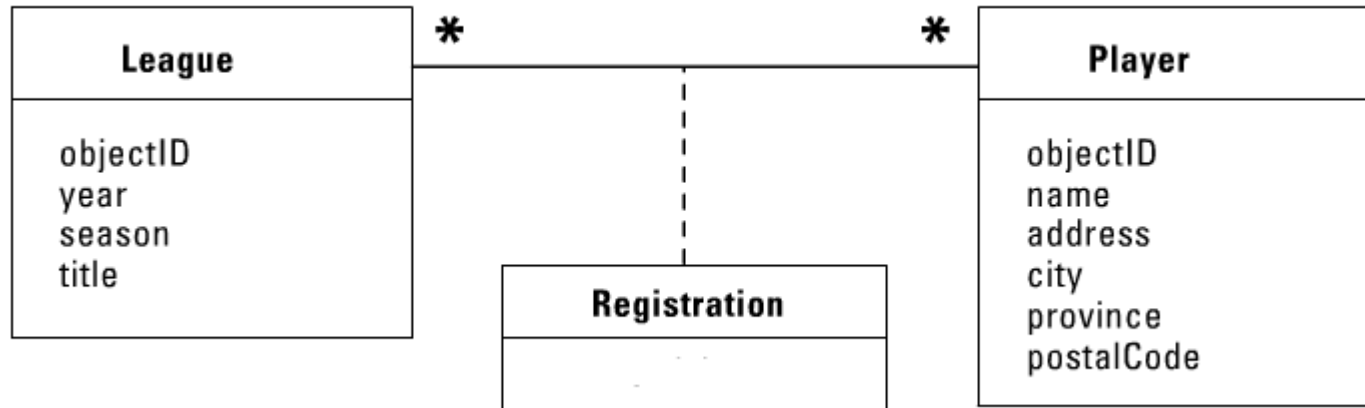
- ObjectNotFoundException

- 領域物件 Domain Object

- Player



# 領域物件對應資料庫資料表



<<table>> League
LID year season title

LID	year	season	title
001	2001	Spring	Soccer League (Spring '01)
002	2001	Summer	Summer Soccer Fest 2001
003	2001	Fall	Fall Soccer League 2001
004	2004	Summer	The Summer of Soccer Love

<<table>> Player
PID name address city province postal_code

PID	name	address	city	province	postal_code
047	Steve Sterling	12 Grove Park Road	Manchester	Manchester	M4 6NF
048	Alice Hornblower	62 Woodside Lane	Reading	Berks	RG31 9TT
049	Wally Winkle	17 Chippenham Road	London	London	SW19 4FT

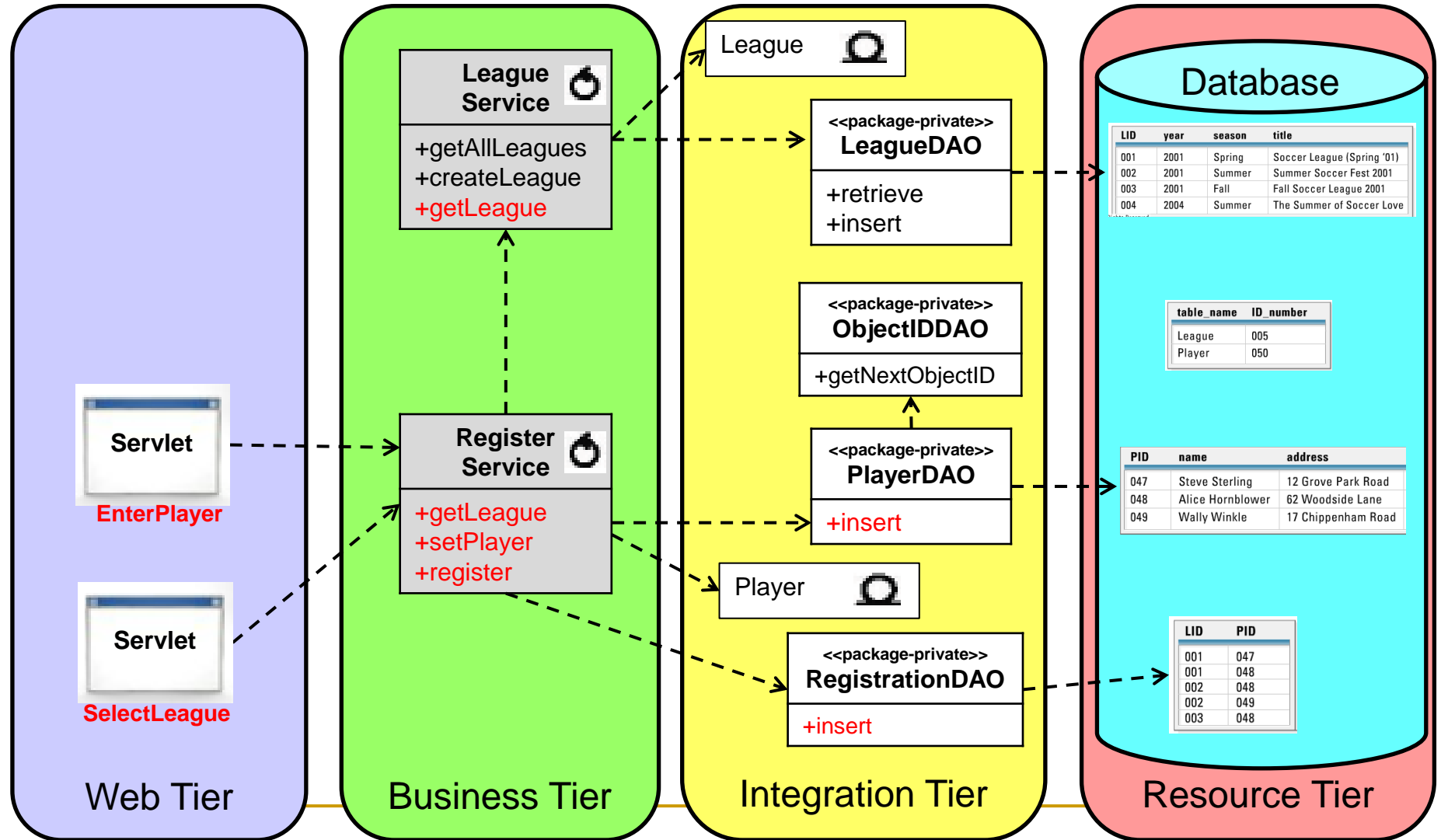
<<table>> ObjectIDs
table_name ID_number

table_name	ID_number
League	005
Player	050

<<table>> Registration
LID PID

LID	PID
001	047
001	048
002	048
002	049
003	048

# Soccer專案實作DBMS整合



# Lab 2 球員註冊功能整合資料庫

## 2. 新增model.PlayerDAO類別

- 宣告新增球員的preparedStatement字串

```
String INSERT_STMT = "INSERT INTO Player (PID, name, address, city, province, postal_code) VALUES (?, ?, ?, ?, ?, ?)";
```

- void insert(Player player) 方法

- 宣告變數：DataSource, Connection, PreparedStatement, ResultSet
- 從JNDI 中取得DataSource
- 從DataSource取得資料庫連線
- 建立一個 INSERT預編敘述
- 設定預編敘述
  - 取得下一個 Player 物件的 object ID, 加入INSERT敘述的第一個欄位
  - 將球員屬性資料加入INSERT敘述的對應欄位
- 執行INSERT敘述
- 例外處理及資源釋放

# Lab 2 球員註冊功能整合資料庫

## 3. 新增model.RegistrationDAO類別

- 宣告新增註冊資料的preparedStatement字串

String **INSERT\_STMT** = "INSERT INTO Registration (LID, PID)  
VALUES (?, ?)";

- void insert(League league, Player player)方法
  - 宣告變數：DataSource, Connection, PreparedStatement, ResultSet
  - 從JNDI 中取得DataSource
  - 從DataSource取得資料庫連線
  - 建立一個 INSERT預編敘述
  - 設定預編敘述
    - 將聯盟物件 objectID 屬性加入INSERT敘述的第一個欄位
    - 將球員物件 playerId 屬性加入INSERT敘述的第二個欄位
  - 執行INSERT敘述
  - 例外處理及資源釋放

# Lab 2 球員註冊功能整合資料庫

## 4. 修改model.LeagueDAO類別

- 新增 League retrieve(int year, String season)方法
  - 依指定年及季,查詢聯盟資料表, 將查詢結果封裝為聯盟物件傳回
  - 聯盟不存在傳回ObjectNotFoundException

## 5. 新增 ObjectNotFoundException類別

## 6. 修改 model.LeagueService類別

- 新增 public League getLeague(int year, String season)方法
  - LeagueDAO的retrieve(year, season)方法取得聯盟物件並傳回
  - 聯盟不存在傳回ObjectNotFoundException

# Lab 2 球員註冊功能整合資料庫

## 7. 修改 `model.RegistrationService`類別

- ❑ 無參數建構子
- ❑ `public League getLeague(int year, String season)`方法
  - `LeagueService`的`getLeague(year, season)`取得聯盟物件並傳回
- ❑ `public void setPlayer(Player player)`
  - 建立`PlayerDAO`物件,將球員資料經`PlayerDAO`物件寫入資料庫
- ❑ `public void register( League league, Player player)`
  - 建立`RegistrationDAO`物件,呼叫`insert()`方法將資料寫入資料庫
- ❑ 移除原本的屬性、方法、建構子

## 8. 修改 `model.Player`類別

- ❑ 增加屬性 `int playerId`
- ❑ 修改建構子, 加入`playerID`傳入參數

```
public Player(int playerId, String name, String address, String city,
String province, String postalCode)
```

# Lab 2 球員註冊功能整合資料庫

## 9. 修改 controller.EnterPlayer類別

- ❑ Player建構子已變更, 多加一個playerID傳入參數
  - 先以-1帶入
  - PlayerDAO會以ObjectIdDAO取的值代換

## 10. 修改 controller.SelectLeague類別

- ❑ 刪除 private League findLeague(int year, String season) 方法
- ❑ 聯盟物件由RegistrationService getLeague()方法取得
- ❑ Session中取得Player物件
- ❑ RegistrationService設定setPlayer(Player player)
- ❑ RegistrationService註冊register(league, player)



# PlayerDAO元

PlayerDAO
#PlayerDAO()
# insert(player : Player)

.....

```
class PlayerDAO {
```

```
    PlayerDAO() { }
```

```
    private static final String INSERT_STMT = "INSERT INTO Player " +  
        " (PID, name, address, city, province, postal_code) VALUES " +  
        " (?, ?, ?, ?, ?, ?)";
```

```
    void insert(Player player) {
```

```
        // JDBC變數
```

```
    .....
```

```
    try {
```

```
        // 從JNDI 中取得DataSource
```

```
    .....
```

```
        // 取得資料庫連線
```

```
        connection = ds.getConnection();
```

```
        // 建立一個 INSERT敘述
```

```
        insert_stmt = connection.prepareStatement(INSERT_STMT);
```

```
        // 取得下一個 Player 物件的 object ID
```

```
        int playerId = ObjectIdDAO.getNextObjectID
```

```
            (ObjectIdDAO.PLAYER, connection);
```

```
        //將取得之 object ID 加入INSERT敘述的第一個欄位
```

```
        insert_stmt.setInt(1, playerId);
```

```
        // 將球員屬性資料加入INSERT敘述的對應欄位
```

```
        insert_stmt.setString(2, player.name);
```

```
    .....
```

```
        insert_stmt.setString(6, player.postalCode);
```

```
        // 執行INSERT敘述
```

```
        insert_stmt.executeUpdate();
```

```
        // 設定球員物件的 object ID
```

```
        player.playerID = playerId;
```

```
        // 例外處理及釋放 JDBC 資源
```

```
    .....
```

# RegistrationDAO元件

RegistrationDAO
-RegistrationDAO()
# insert(league : League, player : Player)

```
.....
class RegistrationDAO {
    RegistrationDAO() { }
    private static final String INSERT_STMT =
        "INSERT INTO Registration (LID, PID) VALUES (?, ?)";
    void insert(League league, Player player) {
        // JDBC變數
        .....
        try {
            // 從JNDI 中取得DataSource
            .....
            // 取得資料庫連線
            connection = ds.getConnection();
            // 建立一個 INSERT敘述
            insert_stmt = connection.prepareStatement(INSERT_STMT);
            // 將聯盟物件 objectID 屬性加入INSERT敘述的第一個欄位
            insert_stmt.setInt(1, league.objectID);
            // 將球員物件 playerId 屬性加入INSERT敘述的第二個欄位
            insert_stmt.setInt(2, player.playerID);
            // 執行INSERT敘述
            insert_stmt.executeUpdate();
            // 例外處理及釋放 JDBC 資源
            .....
        }
    }
}
```

# LeagueDAO元件

## LeagueDAO

#LeagueDAO()

#retrieve(year: int, season: String) : League

#retrieveAll() : List <League>

#insert(league : League)

```
private static final String RETRIEVE_STMT
```

```
= "SELECT * FROM League WHERE yearno=? AND season=?";
```

League retrieve(int year, String season) throws

ObjectNotFoundException {

```
// JDBC變數
```

```
DataSource ds = null;
```

```
Connection connection = null;
```

```
PreparedStatement stmt = null;
```

```
ResultSet results = null;
```

```
int num_of_rows = 0;
```

```
// 領域物件變數
```

```
League league = null;
```

```
try {
```

```
    // 從JNDI 中取得DataSource
```

```
    Context ctx = new InitialContext();
```

```
    if ( ctx == null ) {
```

```
        throw new RuntimeException("JNDI Context could not be found.");
```

```
    }
```

```
    ds = (DataSource)ctx.lookup("java:comp/env/jdbc/leagueDB");
```

```
    if ( ds == null ) {
```

```
        throw new RuntimeException("DataSource could not be found.");
```

```
    }
```

```
// 取得資料庫連線
```

```
connection = ds.getConnection();
```

# LeagueDAO元件

## LeagueDAO

#LeagueDAO()

#retrieve(year: int, season: String) : League

#retrieveAll() : List <League>

#insert(league : League)

```
// 建立一個SELECT 敘述
stmt = connection.prepareStatement(RETRIEVE_STMT);
// 初始化該敘述並執行查詢動作
stmt.setInt(1, year);
stmt.setString(2, season);
results = stmt.executeQuery();
// 迭代查詢的結果
while ( results.next() ) {
    int objectID = results.getInt("LID");
    // 應該只有一筆資料會回傳
    num_of_rows++;
    if ( num_of_rows > 1 ) {
        throw new SQLException("Too many rows were returned.");
    }
    // 建立並填寫League 物件中的資料
    league = new League(objectID, results.getInt("yearno"),
        results.getString("season"), results.getString("title"));
}
if ( league != null ) {
    return league;
} else {
    throw new ObjectNotFoundException();
}
```

# LeagueDAO 元件

## ObjectNotFoundException 例外

### LeagueDAO

#LeagueDAO()

#retrieve(year: int, season: String) : League

#retrieveAll() : List <League>

#insert(league : League)

```
} catch (SQLException se) { // 處理 SQL 錯誤
    throw new RuntimeException("A database error: " + se.getMessage());
} catch (NamingException ne) { // 處理 JNDI 錯誤
    throw new RuntimeException("A JNDI error: " + ne.getMessage());
} finally { // 釋放 JDBC 資源
    if ( results != null ) {
        try { results.close(); } catch (SQLException se) { se.printStackTrace(); }
    }
    if ( stmt != null ) {
        try { stmt.close(); } catch (SQLException se) { se.printStackTrace(); }
    }
    if ( connection != null ) {
        try { connection.close(); } catch (Exception e) { e.printStackTrace(); }
    }
} // END of try-catch-finally block
} // END of the retrieve method
```

### ObjectNotFoundException

package model;

```
public class ObjectNotFoundException extends Exception {
}
```

# LeagueService

## LeagueService

+LeagueService()

+createLeague(year : int, season : String, title: String)

+getAllLeague():List<League>

+getLeague(year : int, season : String):League

```
package model;
// 引入Utility 類別
import java.util.List;
public class LeagueService {
    //宣告服務元件所使用之LeagueDAO物件
    private LeagueDAO leagueDataAccess;
    //建構子, 初始化LeagueDAO物件
    public LeagueService() {
        leagueDataAccess = new LeagueDAO();
    }
    //取得所有聯盟.使用LeagueDAO
    public List<League> getAllLeagues() {
        return leagueDataAccess.retrieveAll();
    }
    //新增聯盟至資料庫
    public League createLeague(int year, String season, String title) {
        // 建立聯盟物件,ObjectID未知,以-1帶入
        League league = new League(-1, year, season, title);
        //新增聯盟至資料庫
        leagueDataAccess.insert(league);
        return league;
    }
    //依指定年份及季節資料,傳回對應聯盟物件,聯盟不存在傳回例外
    public League getLeague(int year, String season)
        throws ObjectNotFoundException {
        return leagueDataAccess.retrieve(year, season);
    }
}
```

# RegisterService元件

RegisterService
+RegisterService()
+getLeague(year : int, season : String) +getPlayer(name : String) +register(league : League, player: Player)

```
package model;
```

```
public class RegisterService {
```

```
    public RegisterService() { }
```

```
    public League getLeague(int year, String season)
        throws ObjectNotFoundException {
```

```
        // 將工作委派給 league service
```

```
        LeagueService leagueSvc = new LeagueService();
```

```
        return leagueSvc.getLeague(year, season);
```

```
    }
```

```
    public void setPlayer(Player player){
```

```
        // 建立PlayerDAO物件,將球員資料寫入資料庫
```

```
        PlayerDAO playerDataAccess = new PlayerDAO();
```

```
        playerDataAccess.insert(player);
```

```
    }
```

```
    public void register(League league, Player player) {
```

```
        // 建立RegistrationDAO物件,將註冊資料寫入資料庫
```

```
        RegistrationDAO registrator = new RegistrationDAO();
```

```
        registrator.insert(league, player);
```

```
    }
```

```
}
```

# Player.Java

```
package model;

public class Player {
    int playerID;
    String name;
    String address;
    String city;
    String province;
    String postalCode;

    public Player(int playerID, String name, String address, String city, String province, String postalCode) {
        this.playerID = playerID;
        this.name = name;
        this.address = address;
        this.city = city;
        this.province = province;
        this.postalCode = postalCode;
    }

    .....
}
```



# EnterPlayer.Java

```
package controller;

.....
@WebServlet(name = "EnterPlayer", urlPatterns = {"EnterPlayer"})
public class EnterPlayer extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse
                                response) throws ServletException, IOException {

        .....
        // Retrieve form parameters.
        // Verify form parameters

        .....
        // Send the ErrorPage if there were errors

        .....
        // Perform business logic
        Player player = new Player(-1, name, address, city, province, postalCode);
        HttpSession session = request.getSession();
        session.setAttribute("player", player);

        // Send the Success page

        .....
    }
}
```

# SelectLeague.Java

```
package controller;

.....

@WebServlet(name = "SelectLeague", urlPatterns = {"SelectLeague"})
public class SelectLeague extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse
                                response) throws ServletException, IOException {

        .....

        // Retrieve form parameters.
        // Verify form parameters
        RegistrationService registration = new RegistrationService();
        League league = null;
        try {
            league = registration.getLeague(year, season);
        } catch (ObjectNotFoundException oe){
            errorMsgs.add("Please enter valid league.");
        }

        // Send the ErrorPage if there were errors
        // Perform business logic
        HttpSession session = request.getSession();
        session.setAttribute("league", league);
        Player player = (Player)session.getAttribute("player");
        registration.setPlayer(player);
        registration.register(league, player);

        // Send the Success page
        .....

    }
}
```

Duke's Soccer League: Registration

1) Enter Player Info   2) Select League

Name:

Address:

City:

Province:

Postal code:

Duke's Soccer League: Registration

1) Enter Player Info   2) Select League

Year:

Season:

Duke's Soccer League: Registration

1) Enter Player Info   2) Select League

Please correct the following errors:

- Please enter valid league.

Year:

Season:

SELECT \* FROM SPUB.LEAGUE... x

Max. rows: 100 | Fetched Rows: 8 |

#	LID	YEARNO	SEASON	TITLE
1		1	2019 Spring	Soccer League (Spring '19)
2		2	2019 Summer	Summer Soccer Fest 2019
3		3	2019 Fall	Fall Soccer League (2019)
4		4	2020 Spring	Soccer League (Spring '20)
5		5	2020 Summer	The Summer of Soccer Love 2020
6		6	2020 Fall	Fall Soccer League (2020)
7		7	2020 Winter	2020 Winter Game

Duke's Soccer League: Registration

1) Enter Player Info2) Select League

Name:

Address:

City:

Province:

Postal code:

Continue...

SELECT \* FROM SPUB.PLAYER...

#	PID	NAME	ADDRESS	CITY	PROVINCE	POSTAL_CODE
1	1	Sean	3F, 30, Park Road	Taipei	Taiwan	100

SELECT \* FROM SPUB.REGIST...

#	LID	PID
1		7

SELECT \* FROM SPUB.OBJECT...

#	TABLE_NAME	ID_NUMBER
1	League	8
2	Player	2

Duke's Soccer League: Registration

1) Enter Player Info2) Select League

Please correct the following errors:

- Please enter valid league.

Year:

Season:

Submit

Duke's Soccer League: Registration Success

Thank you, Sean.  
Your request to register the 2020 Winter Game league was successful.

[Register another league](#)  
[Return to homepage](#)