Database Principle Assignment Report

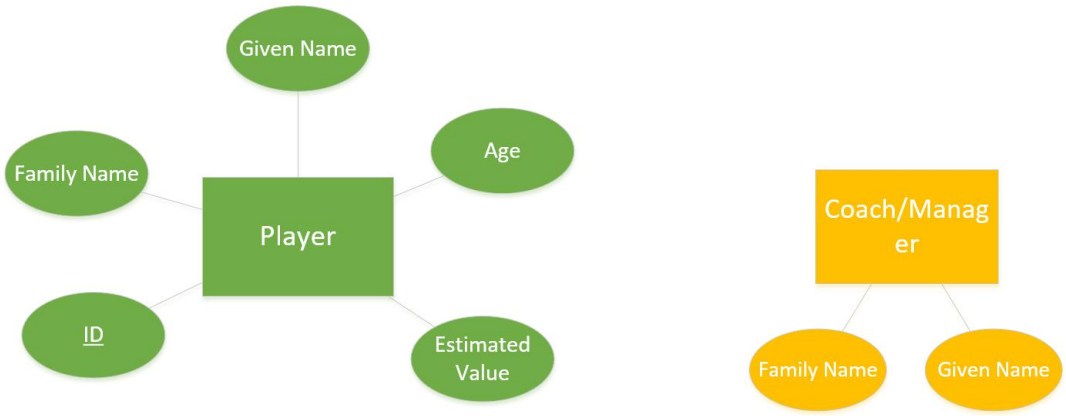
	Student 1	Student 2
Student Name:	唐欣然	陈铭杰
English Name:	Xinran Tang	Mingjie Chen
Student ID:	20175345	20175447

May 2019

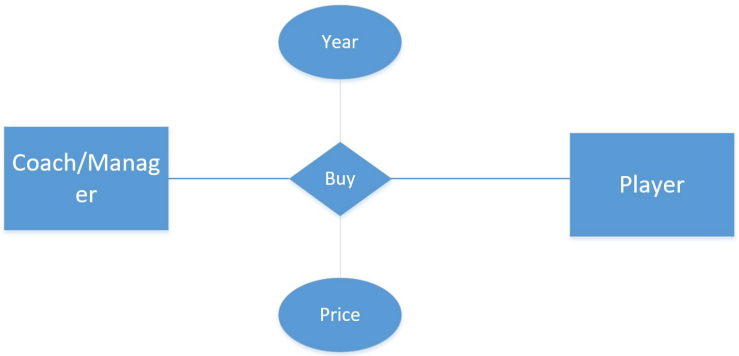
SECTION 1

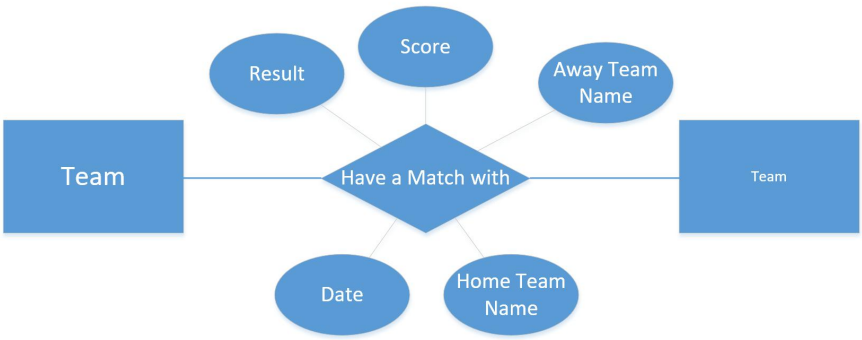
ER DIAGRAM

1. Entities

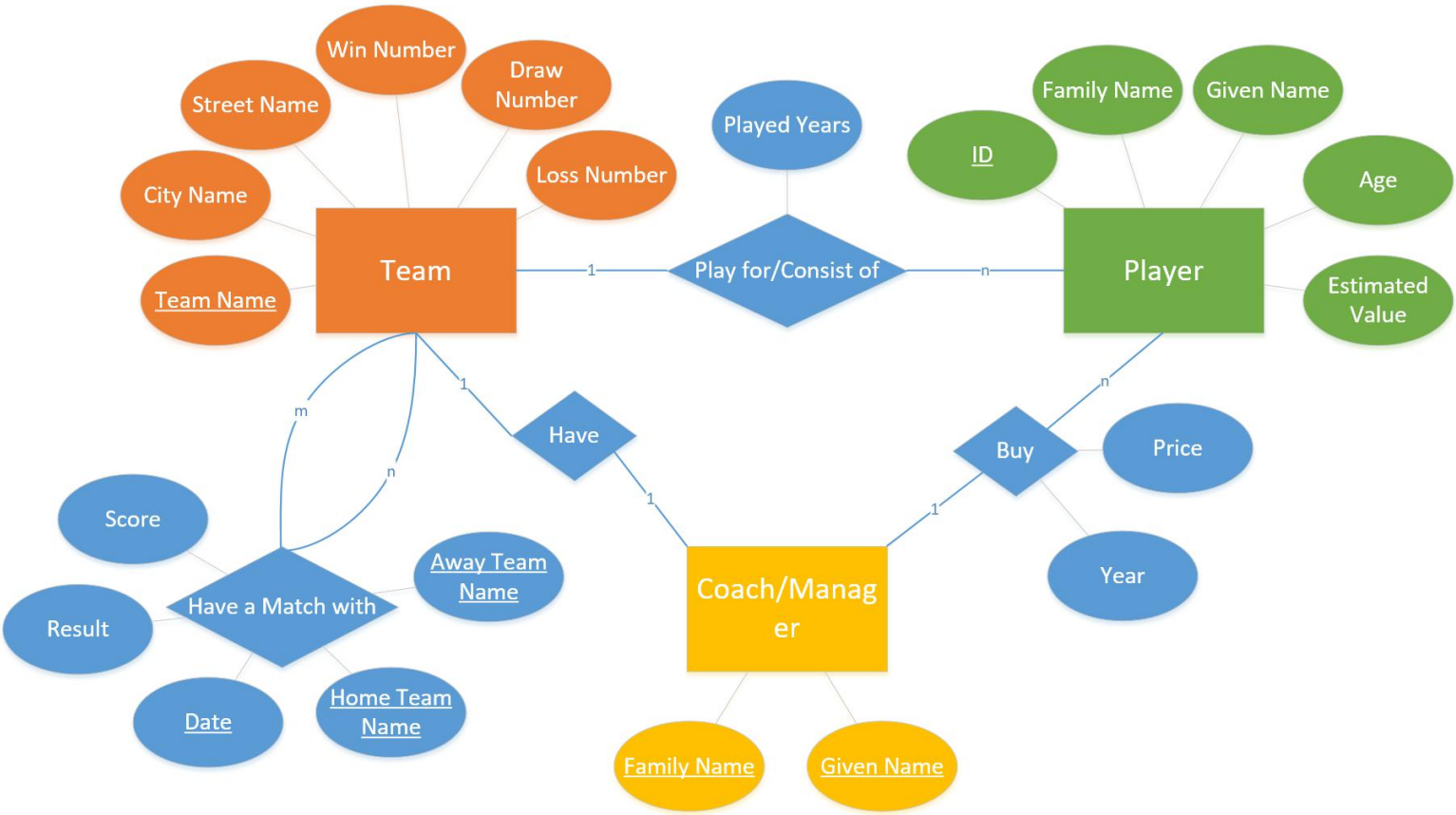


2. Relation between entities





3. ER Diagram


Win Number
Draw Number
Street Name
Played Years
Family Name
Given Name
ID
Age
City Name
Loss Number
Team
1
Play for/Consist of
n
Player
Team Name
Estimated Value
m
1
n
Have
Buy
Price
Score
n
1
1
Away Team Name
Have a Match with
Coach/Manager
Year
Result
Date
Home Team Name
Family Name
Given Name

SECTION 2

NORMALISATION BY DECOMPOSITION

- **Original:**

➢ R:

team_name	city_name	street_name	win_number	draw_number	loss_number
played_year	player_ID	p_family_name	p_given_name	age	estimated_value
price	buy_year	c_family_name	c_given_name	score	result
date	home_team_name	Away_team_name			

➢ FD: {team_name->city_name, street_name, win_number, draw_number, loss_number
playerID, team_name -> played_years;
playerID -> p_family_name, p_given_name, age, estimated_value;
playerID, c_family_name, c_given_name -> price, year;
home_team_name, away_team_name, data -> result, score}

- **R calculate the key**

➢ R (×)

team_name	city_name	street_name	win_number	draw_number	loss_number
played_year	player_ID	p_family_name	p_given_name	age	estimated_value
price	buy_year	c_family_name	c_given_name	score	result
date	home_team_name	Away_team_name			

- **Decompose R on home_team_name, away_team_name, date -> result,score**

➢ R1

away_team_name	home_team_name	date	score	result

➢ FD: {home_team_name, away_team_name, date -> result,score}

➢ R2 (×)

team_name	city_name	street_name	win_number	draw_number	loss_number

played_year	player_ID	p_family_name	p_given_name	age	estimated_value

price	buy_year	c_family_name	c_given_name

data	home_team_name	away_team_name

➢ FD:{team_name -> city_name, street_name, win_number, draw_number;
player_ID, team_name -> played_years;
player_ID -> p_family_name, p_given_name, price, year}

- **Decompose R2 on player_ID-> c_family_name, c_given_name, price, year**

➢ R3

player_ID	c_family_name	c_given_name	price	buy_year

➢ FD: {player_ID, c_family_name, c_given_name -> price, year}

- R4 (×)

team_name	city_name	street_name	win_number	draw_number	loss_number
played_year	player_ID	p_family_name	p_given_name	age	estimated_value
c_family_name	c_given_name	date	home_team_name	away_team_name	

- FD: {team_name -> city_name, street_name, win_number, draw_number;
  player_ID, team_name -> played_years;
  player_ID -> p_family_name, p_given_name, age, estimated_value}

• Decompose R4 on player_ID -> p_family_name, p_given_name, age, estimated_value

- R5

player_ID	p_family_name	p_given_name	age	estimated_value

- FD:{player_ID -> p_family_name, p_given_name, age, estimated_value}

- R6 (×)

team_name	city_name	street_name	win_number	draw_number	loss_number
played_year	player_ID	c_family_name	c_given_name		
home_team_name	home_team_name	away_team_name			

- FD: {team_name -> city_name, street_name, win_number, draw_number;
  player_ID, team_name -> played_years}

• Decompose R6 on player_ID, team_name -> played_years

- R7

team_name	player_ID	played_years

- FD: {player_ID, team_name -> played_years}

- R8 (×)

team_name	city_name	street_name	win_number	draw_number	loss_number

player_ID	c_family_name	c_given_name

home_team_name	home_team_name	away_team_name

- FD: {team_name -> city_name, street_name, win_number, draw_number}

- **Decompose R8 on team_name -> city_name, street_name, win_number, draw_number**

- R9

team_name	city_name	street_name	win_number	draw_number	loss_number

- FD: {team_name -> city_name, street_name, win_number, draw_number}

R10

team_name	player_ID	c_family_name	c_given_name

date	home_team_name	away_team_name

- FD:{}

SECTION 3

SQL DDL COMMANDS

1. DDL Codes

```
1.  create database basketball_league;
2.  use basketball_league;
3.  CREATE TABLE Team(
4.      name VARCHAR(50) UNIQUE NOT NULL,
5.      street_name VARCHAR(50),
6.      city_name VARCHAR(50),
7.      win_number INT DEFAULT 0 CHECK (win_number >0),
8.      draw_number INT DEFAULT 0 CHECK (draw_number >0),
9.      loss_number INT DEFAULT 0 CHECK (loss_number >0),
10.     c_family_name VARCHAR(50) NOT NULL,
11.     c_given_name VARCHAR(50) NOT NULL,
12.     PRIMARY KEY (name)
13. );
14.
15. #Team Data
16. Insert Into team values('Hawks', '1 Philips Dr NW.', 'Atlanta', 4, 2, 0, 'Bu
    denholzer', 'Mike');
17. Insert Into team values('Dodgers', 'Academy Road', 'Los Angeles', 3, 0, 2, '
    Roberts', 'Eric');
18. Insert Into team values('Warriors', 'Coliseum Road', 'Oakland', 2, 1, 3, 'Co
    le', 'Steve');
19. Insert Into team values('Clippers', 'South Figueroa Street', 'Los Angeles',
    1, 0, 3, 'Rivers', 'Doug');
20. Insert Into team values('Heat', ' 1 SE Third Ave', 'Miami', 0, 1, 2, 'Spoels
    tra', 'Eric');
21.
22. CREATE TABLE Coach(
23.     c_family_name VARCHAR(50) NOT NULL,
24.     c_given_name VARCHAR(50) NOT NULL,
25.     CONSTRAINT PRIMARY KEY (c_family_name,c_given_name)
26. );
27.
28. #Coach/Manager Data
29. Insert Into Coach values('Budenholzer', 'Mike');
30. Insert Into Coach values('Roberts', 'Eric');
```

```
31. Insert Into Coach values('Cole', 'Steve');
32. Insert Into Coach values('Rivers', 'Doug');
33. Insert Into Coach values('Spoelstra', 'Eric');
34.
35. CREATE TABLE Player(
36.     ID VARCHAR(50) UNIQUE NOT NULL,
37.     p_family_name VARCHAR(50) NOT NULL,
38.     p_given_name VARCHAR(50) NOT NULL,
39.     age INT DEFAULT 18 CHECK (age >= 18 and age <= 40),
40.     estimated_value DOUBLE CHECK(estimated_value>0),
41.     played_years INT CHECK(played_years>0),
42.     team_name VARCHAR(50) NOT NULL,
43.     price DOUBLE CHECK(price>0),
44.     buy_year INT CHECK(buy_year>0),
45.     c_family_name VARCHAR(50) NOT NULL,
46.     c_given_name VARCHAR(50) NOT NULL,
47.     PRIMARY KEY (ID),
48.     FOREIGN KEY (team_name) REFERENCES Team(name)
49.     ON DELETE CASCADE ON UPDATE CASCADE,
50.     CONSTRAINT Player FOREIGN KEY (c_family_name,c_given_name) REFERENCES Co
    ach(c_family_name,c_given_name)
51.     ON DELETE CASCADE ON UPDATE CASCADE
52. );
53.
54. #Player Data
55. Insert Into player values('0101', 'Wade', 'Dwyane', 25, 13000000, 8, 'Hawks',
     3000000, 2011, 'Budenholzer', 'Mike');
56. Insert Into player values('0102', 'Durant', 'Kevin', 23, 12000000, 7, 'Hawks
    ', 2500000, 2012, 'Budenholzer', 'Mike');
57. Insert Into player values('0201', 'James', 'LeBron', 24, 15000000, 9, 'Dodge
    rs', 3500000, 2010, 'Roberts', 'Eric');
58. Insert Into player values('0202', 'DeRozan', 'Demar', 24, 13000000, 7, 'Dodg
    ers', 3500000, 2012, 'Roberts', 'Eric');
59. Insert Into player values('0301', 'Gordon', 'Aaron', 22, 12000000, 4, 'Warri
    ors', 2000000, 2015, 'Cole', 'Steve');
60. Insert Into player values('0302', 'Curry', 'Stephen', 25, 20000000, 6, 'Warr
    iors', 5000000, 2013, 'Cole', 'Steve');
61. Insert Into player values('0401', 'Bryant', 'Kobe', 26, 25000000, 9, 'Clippe
    rs', 6000000, 2010, 'Rivers', 'Doug');
62. Insert Into player values('0402', 'Allen', 'Ray', 23, 18000000, 8, 'Clippers
    ', 5000000, 2011, 'Rivers', 'Doug');
63. Insert Into player values('0501', 'Curry', 'Seth', 24, 11000000, 6, 'Heat',
    2000000, 2013, 'Spoelstra', 'Eric');
```

```
64. Insert Into player values('0502', 'Griffin', 'Blake', 21, 10000000, 3, 'Heat
    ', 1000000, 2016, 'Spoelstra', 'Eric');
65.
66. /*Add Foreign Key to Table Team*/
67. ALTER TABLE Team
68. ADD CONSTRAINT Team FOREIGN KEY (c_family_name,c_given_name) REFERENCES Coac
    h(c_family_name,c_given_name)
69. ON DELETE CASCADE ON UPDATE CASCADE;
70.
71. CREATE TABLE BasketballMatch (
72.     away_team VARCHAR(50) NOT NULL,
73.     home_team VARCHAR(50) NOT NULL,
74.     match_date VARCHAR(50) NOT NULL,
75.     score VARCHAR(50) NOT NULL,
76.     result VARCHAR(50) NOT NULL,
77.     CONSTRAINT BasketballMatch PRIMARY KEY (away_team , home_team , match_da
    te),
78.     FOREIGN KEY (away_team)
79.         REFERENCES Team (name)
80.         ON DELETE CASCADE ON UPDATE CASCADE,
81.     FOREIGN KEY (home_team)
82.         REFERENCES Team (name)
83.         ON DELETE CASCADE ON UPDATE CASCADE
84. );
85.
86. #BasketballMatch Data
87. #result represents whether home team wins or not
88. Insert Into BasketballMatch values('Heat', 'Hawks', '9 Mar 2019', '85:98', '
    lose');
89. Insert Into BasketballMatch values('Hawks', 'Dodgers', '12 Mar 2019', '93:88
    ', 'win');
90. Insert Into BasketballMatch values('Hawks', 'Warriors', '16 Mar 2019', '87:8
    5', 'win');
91. Insert Into BasketballMatch values('Hawks', 'Clippers', '23 Mar 2019', '90:8
    9', 'win');
92. Insert Into BasketballMatch values('Warriors', 'Hawks', '29 Mar 2019', '87:8
    7', 'draw');
93. Insert Into BasketballMatch values('Heat', 'Hawks', '31 Mar 2019', '68:68',
    'draw');
94. Insert Into BasketballMatch values('Dodgers', 'Heat', '2 April 2019', '83:81
    ', 'win');
95. Insert Into BasketballMatch values('Dodgers', 'Warriors', '5 April 2019', '1
    01:97', 'win');
```

```
96. Insert Into BasketballMatch values('Clippers', 'Dodgers', '8 April 2019', '8
    9:92', 'lose');
97. Insert Into BasketballMatch values('Warriors', 'Dodgers', '12 April 2019', '
    99:98', 'win');
98. Insert Into BasketballMatch values('Warriors', 'Clippers', '18 April 2019',
    '97:95', 'win');
99. Insert Into BasketballMatch values('Clippers', 'Warriors', '23 April 2019',
    '94:90', 'win');
```

2. Tables

ID	p_family_name	p_given_name	age	estimated_value	played_years	team_name	price	buy_year	c_family_name	c_given_name
0101	Wade	Dwyane	25	13000000	8	Hawks	3000000	2011	Budenholzer	Mike
0102	Durant	Kevin	23	12000000	7	Hawks	2500000	2012	Budenholzer	Mike
0201	James	LeBron	24	15000000	9	Dodgers	3500000	2010	Roberts	Eric
0202	DeRozan	Demar	24	13000000	7	Dodgers	3500000	2012	Roberts	Eric
0301	Gordon	Aaron	22	12000000	4	Warriors	2000000	2015	Cole	Steve
0302	Curry	Stephen	25	20000000	6	Warriors	5000000	2013	Cole	Steve
0401	Bryant	Kobe	26	25000000	9	Clippers	6000000	2010	Rivers	Doug
0402	Allen	Ray	23	18000000	8	Clippers	5000000	2011	Rivers	Doug
0501	Curry	Seth	24	11000000	6	Heat	2000000	2013	Spoelstra	Eric
0502	Griffin	Blake	21	10000000	3	Heat	1000000	2016	Spoelstra	Eric

Table 1. Players

c_family_name	c_given_name
Budenholzer	Mike
Cole	Steve
Rivers	Doug
Roberts	Eric
Spoelstra	Eric

Table2. Coaches/Managers

away_team	home_team	match_date	score	result
Clippers	Dodgers	8 April 2019	89:92	lose
Clippers	Warriors	23 April 2019	94:90	win
Dodgers	Heat	2 April 2019	83:81	win
Dodgers	Warriors	5 April 2019	101:97	win
Hawks	Clippers	23 Mar 2019	90:89	win
Hawks	Dodgers	12 Mar 2019	93:88	win
Hawks	Warriors	16 Mar 2019	87:85	win
Heat	Hawks	31 Mar 2019	68:68	draw
Heat	Hawks	9 Mar 2019	85:98	lose
Warriors	Clippers	18 April 2019	97:95	win
Warriors	Dodgers	12 April 2019	99:98	win
Warriors	Hawks	29 Mar 2019	87:87	draw

Table3. Matches

team_name	street_name	city_name	win_number	draw_number	loss_number	c_family_name	c_given_name
Clippers	South Figueroa Street	Los Angeles	1	0	3	Rivers	Doug
Dodgers	Academy Road	Los Angeles	3	0	2	Roberts	Eric
Hawks	1 Philips Dr NW.	Atlanta	4	2	0	Budenholzer	Mike
Heat	1 SE Third Ave	Miami	0	1	2	Spoelstra	Eric
Warriors	Coliseum Road	Oakland	2	1	3	Cole	Steve

Table4. Teams

SECTION 4

USER QUERIES(DML)

The attribute ‘result’ in table ‘BasketballMatch’ represents whether home team win or lose

1.show the score table for this year's points

```
1.  select team_name as Name, (win_number+draw_number+loss_number) as Played, win_number a
    s Win,draw_number as Draw, loss_number as Loss, (win_number*2+draw_number) as Points
2.  from Team;
```

	Name	Played	Win	Draw	Loss	Points
▶	Clippers	4	1	0	3	2
	Dodgers	5	3	0	2	6
	Hawks	6	4	2	0	10
	Heat	3	0	1	2	1
	Warriors	6	2	1	3	5

2.show the score table for this year's and next year's points

```
1.  select team_name as Name, (win_number+draw_number+loss_number) as Played, win_number a
    s Win,draw_number as Draw, loss_number as Loss, (win_number*2+draw_number) as Points, (win
    _number*3+draw_number*2-loss_number*1) as 'Next Year\'s Points'
2.  from Team;
```

	Name	Played	Win	Draw	Loss	Points	Next Year's Points
▶	Clippers	4	1	0	3	2	0
	Dodgers	5	3	0	2	6	7
	Hawks	6	4	2	0	10	16
	Heat	3	0	1	2	1	0
	Warriors	6	2	1	3	5	5

3.Look for a match using away team name, home team name and date

```
1.  select * from basketballmatch
2.  where away_team = 'Clippers' and home_team = 'Warriors' and match_date = '23 April 2019';
```

	home_team	away_team	match_date	score	result
▶	Warriors	Clippers	23 April 2019	94:90	win

4.Look for all the matches on one day

```
1.  select * from basketballmatch
```

```
2.  where match_date = '12 April 2019';
```

	home_team	away_team	match_date	score	result
▶	Dodgers	Warriors	12 April 2019	99:98	win

5.Look for all the matches between two teams

```
1.  select * from basketballmatch
2.  where (away_team = 'Clippers' and home_team = 'Warriors') or (away_team = 'Warriors' and home
    _team = 'Clippers');
```

	home_team	away_team	match_date	score	result
▶	Warriors	Clippers	23 April 2019	94:90	win
	Clippers	Warriors	18 April 2019	97:95	win

6.Look for attributes of a player bys ID

```
1.  select * from player
2.  where ID = '0202';
```

	ID	p_family_name	p_given_name	age	estimated_value	played_years	team_name	price	buy_year	c_family_name	c_given_name
▶	0202	DeRozan	Demar	24	13000000	7	Dodgers	3500000	2012	Roberts	Eric

7.Look for all players with the same family name

```
1.  select p1.ID as ID_1,p1.p_family_name as p_family_name_1,p1.p_given_name as p_given_name_1,p
    1.team_name as team_1 ,p2.ID as ID_2,p2.p_family_name as family_name_2,p2.p_given_name as giv
    en_name_2 ,p2.team_name as team_2 from player p1,player p2
2.  where p1.p_family_name = p2.p_family_name and p1.ID<p2.ID;
```

	ID_1	p_family_name_1	p_given_name_1	team_1	ID_2	family_name_2	given_name_2	team_2
▶	0302	Curry	Stephen	Warriors	0501	Curry	Seth	Heat

8.Look for players that works for a certain team by a team name

```
1.  select ID,p_family_name,p_given_name from player
2.  where team_name = 'Clippers';
```

	ID	p_family_name	p_given_name
▶	0401	Bryant	Kobe
	0402	Allen	Ray

9.According to coach name, look for his players buying year & price

```
1.  select ID,p_family_name,p_given_name,price, buy_year from player
2.  where c_family_name = 'Cole' and c_given_name = 'Steve';
```

	ID	p_family_name	p_given_name	price	buy_year
▶	0301	Gordon	Aaron	2000000	2015
	0302	Curry	Stephen	5000000	2013

10.Look for a team's manager/coach and his attributes

```
1.  select c_family_name as 'family name' , c_given_name as 'given name' from te
    am
2.  where team_name = 'Heat';
```

	family name	given name
▶	Spoelstra	Eric

11.Look for the youngest player in a team

```
1.  select p_family_name,p_given_name ,age
2.  from player
3.  where team_name = 'Hawks' and age <= all(select age from player where team_n
    ame = 'Hawks');
```

	p_family_name	p_given_name	age
▶	Durant	Kevin	23

12.Get the average age for every team

```
1.  select team_name,avg(age)
2.  from player
3.  group by team_name;
```

	team_name	avg(age)
▶	Clippers	24.5000
	Dodgers	24.0000
	Hawks	24.0000
	Heat	22.5000
	Warriors	23.5000

13.Look for players bought by a coach/manager order by prices in descending order

```
1.  select p_family_name,p_given_name, price
2.  from player
3.  where c_family_name = 'Rivers' and c_given_name = 'Doug'
4.  order by price desc;
```

	p_family_name	p_given_name	price
▶	Bryant	Kobe	6000000
	Allen	Ray	5000000

14.Look for a team's players whose age is within a range

```
1.  select p_family_name,p_given_name,age
2.  from player
3.  where team_name = 'Heat' and age between 20 and 23;
```

	p_family_name	p_given_name	age
▶	Griffin	Blake	21

15.Look for all of a team's matches as home team

```
1.  select * from basketballmatch
2.  where home_team = 'Dodgers';
```

	home_team	away_team	match_date	score	result
▶	Dodgers	Clippers	8 April 2019	89:92	lose
	Dodgers	Hawks	12 Mar 2019	93:88	win
	Dodgers	Warriors	12 April 2019	99:98	win

16.Look for all the teams in one city

```
1.  select team_name,street_name ,win_number,draw_number,loss_number from team
2.  where team.city_name = 'Los Angeles';
```

	team_name	street_name	win_number	draw_number	loss_number
▶	Clippers	South Figueroa Street	1	0	3
	Dodgers	Academy Road	3	0	2

17.Look for all satisfied players that has Estimated Value within a range

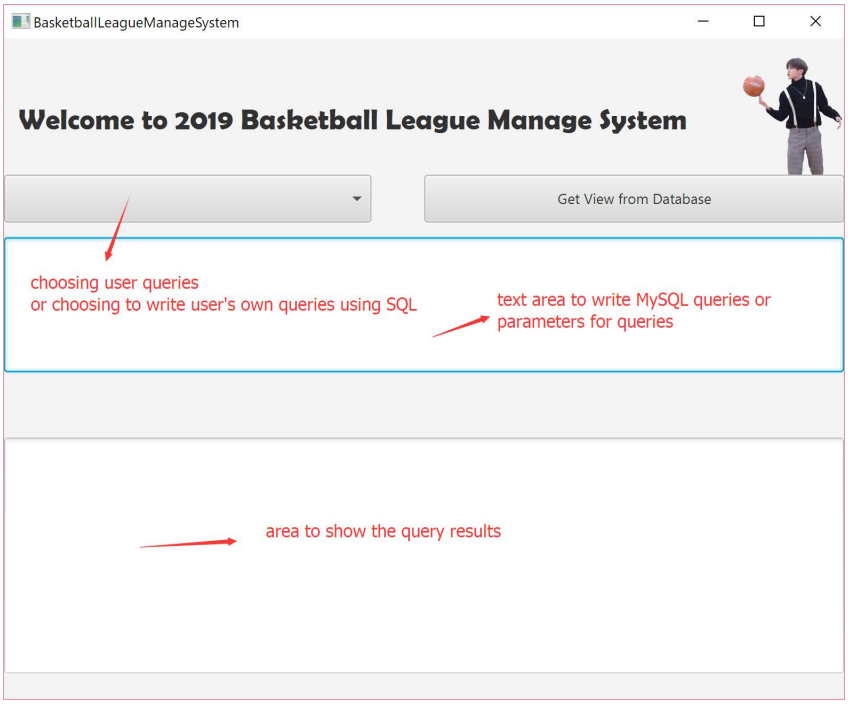
```
1.  select ID, p_family_name,p_given_name,team_name from player
2.  where estimated_value between 12000000 and 18000000;
```

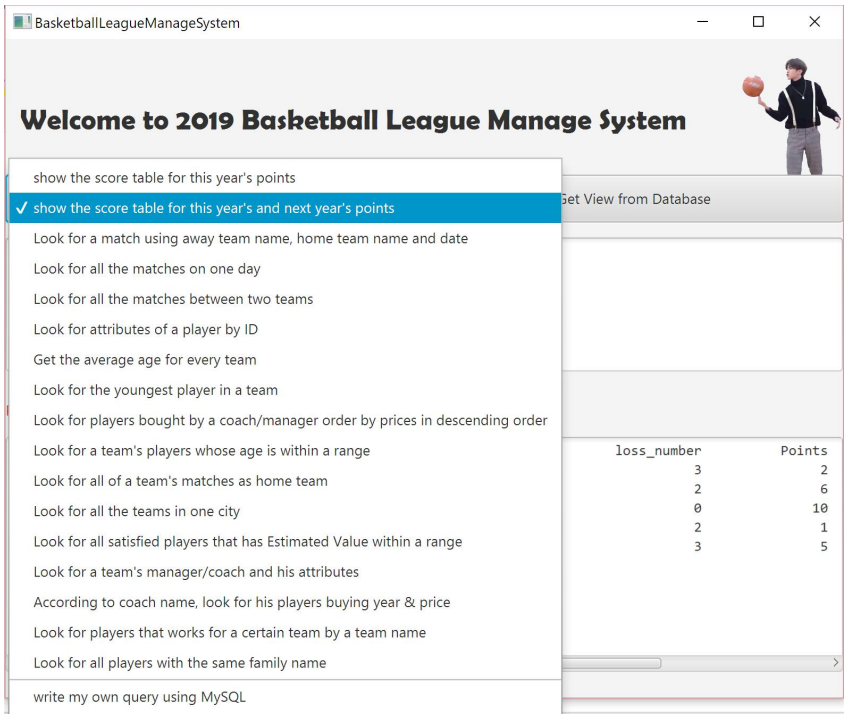
	ID	p_family_name	p_given_name	team_name
▶	0101	Wade	Dwyane	Hawks
	0102	Durant	Kevin	Hawks
	0201	James	LeBron	Dodgers
	0202	DeRozan	Demar	Dodgers
	0301	Gordon	Aaron	Warriors
	0402	Allen	Ray	Clippers

SECTION 5

User Queries Using JAVA UI AND JDBC

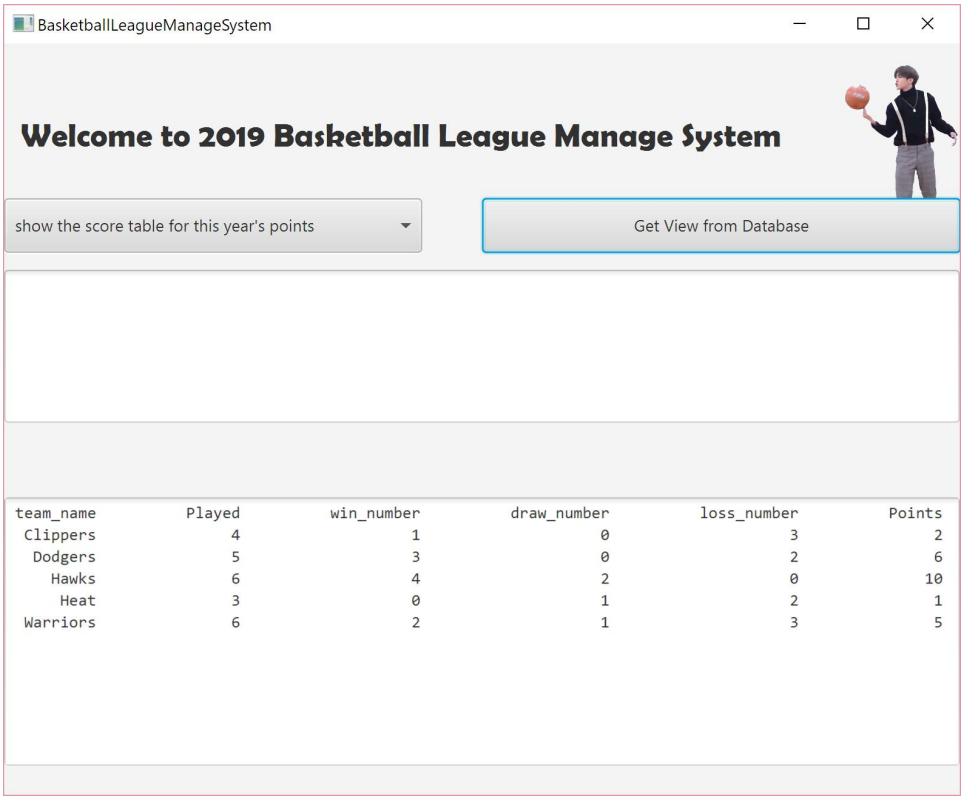
1. User Interface Using JavaFX





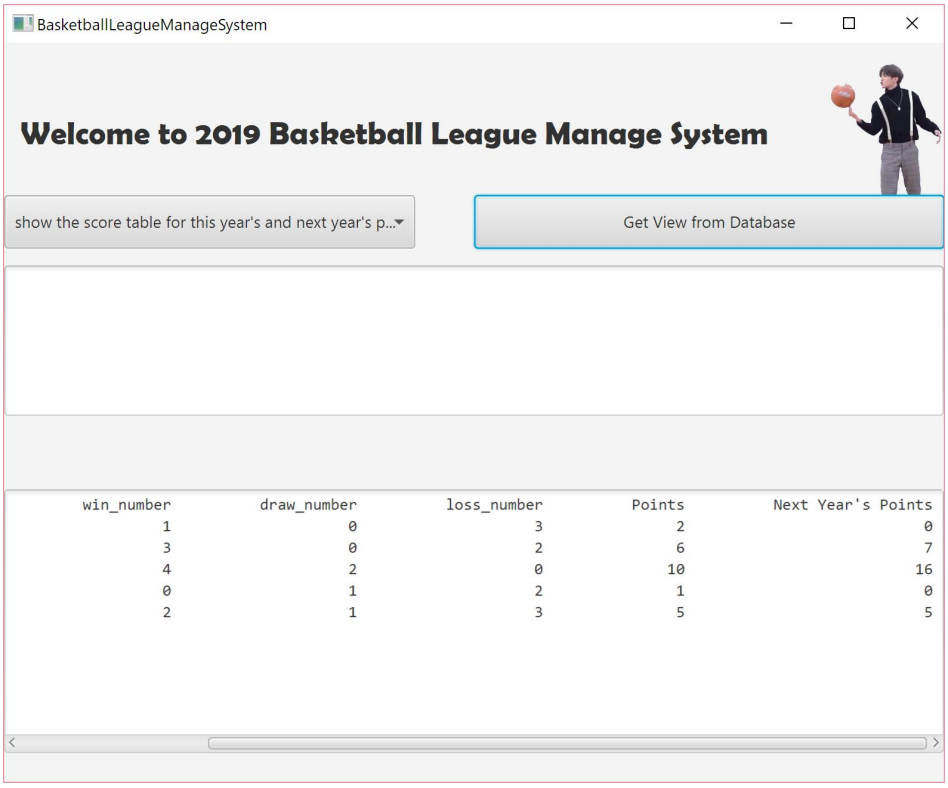
2. Get views from user interface (Part of the samples)

2.1.show the score table for this year's points



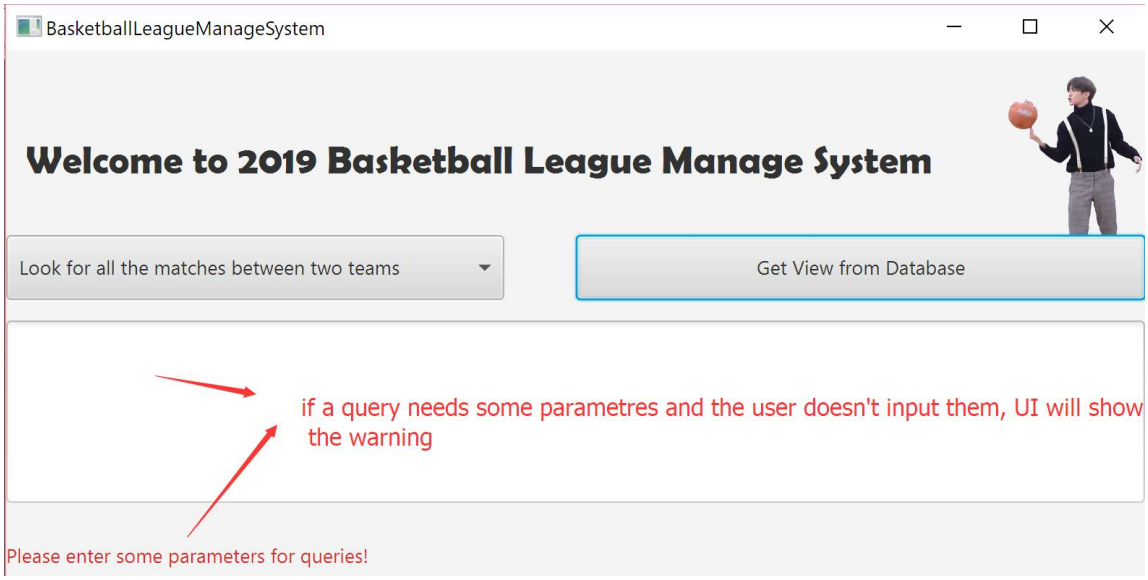
team_name	Played	win_number	draw_number	loss_number	Points
Clippers	4	1	0	3	2
Dodgers	5	3	0	2	6
Hawks	6	4	2	0	10
Heat	3	0	1	2	1
Warriors	6	2	1	3	5

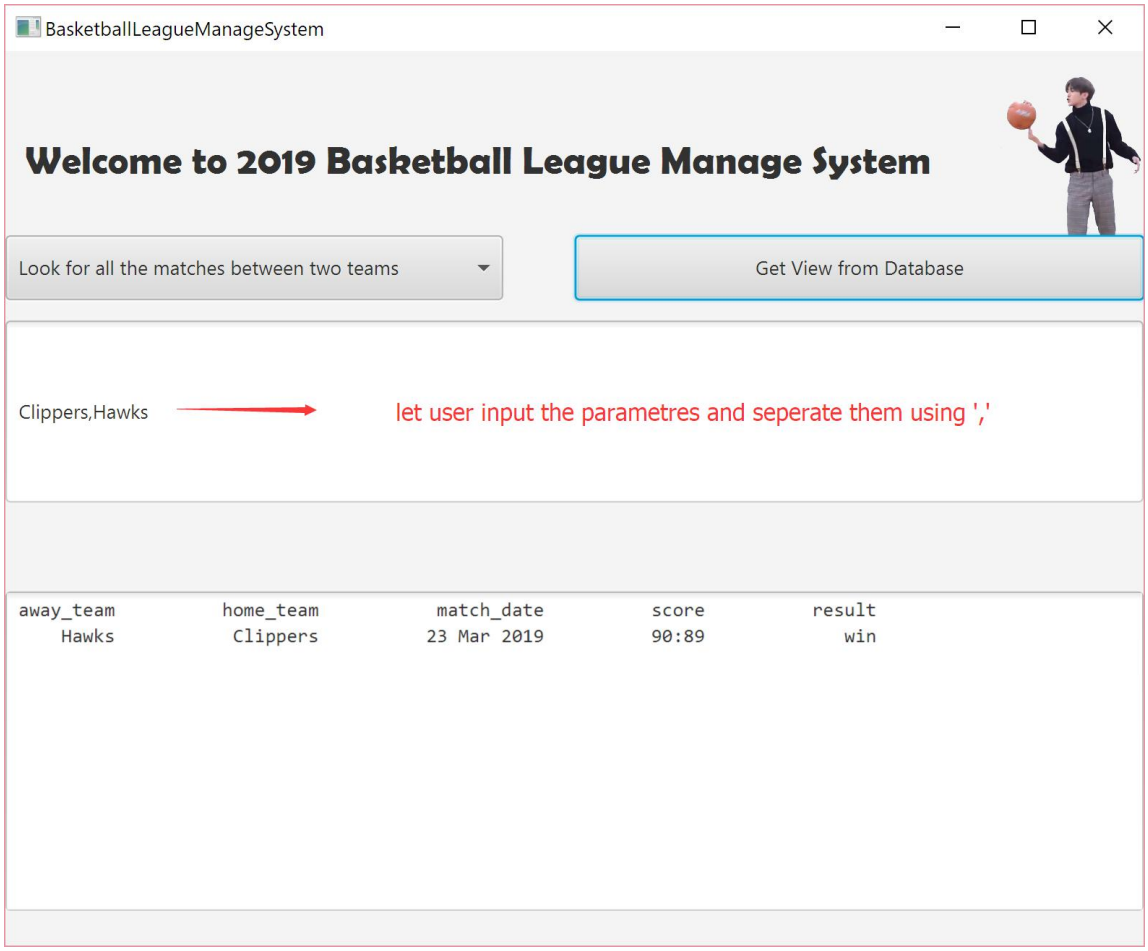
2.2.show the score table for this year's and next year's points



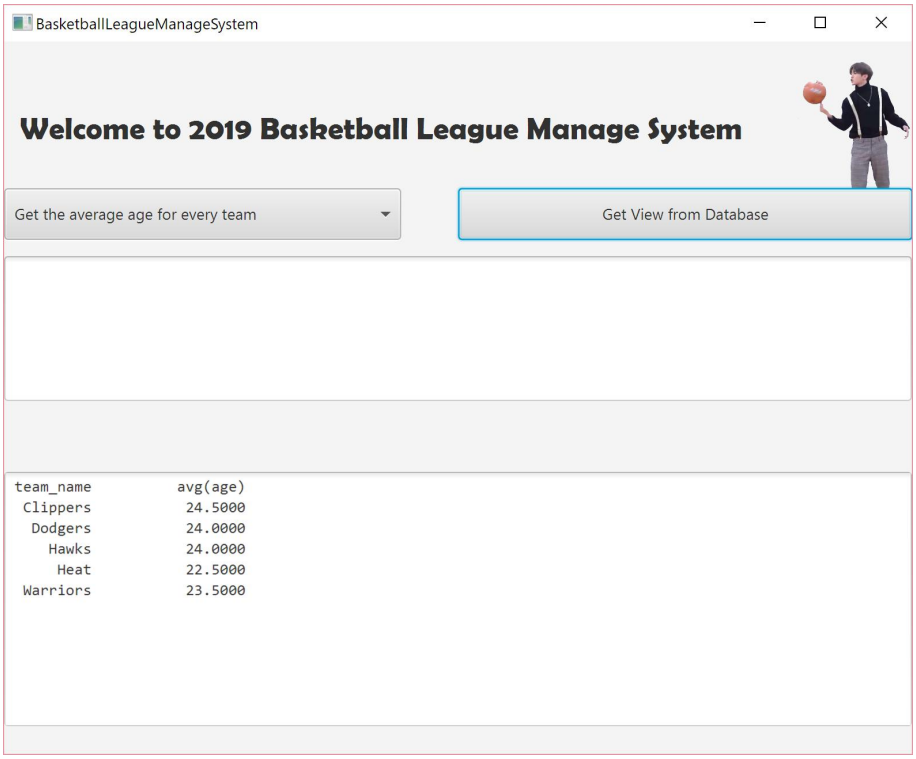
win_number	draw_number	loss_number	Points	Next Year's Points
1	0	3	2	0
3	0	2	6	7
4	2	0	10	16
0	1	2	1	0
2	1	3	5	5

2.3.Look for a match using away team name, home team name and date

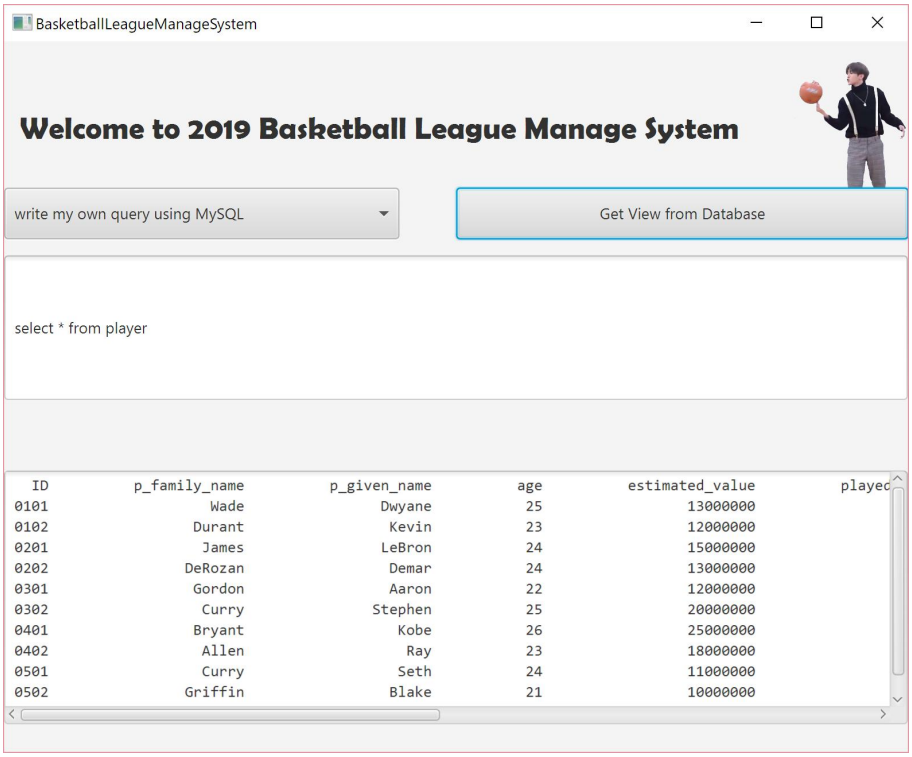


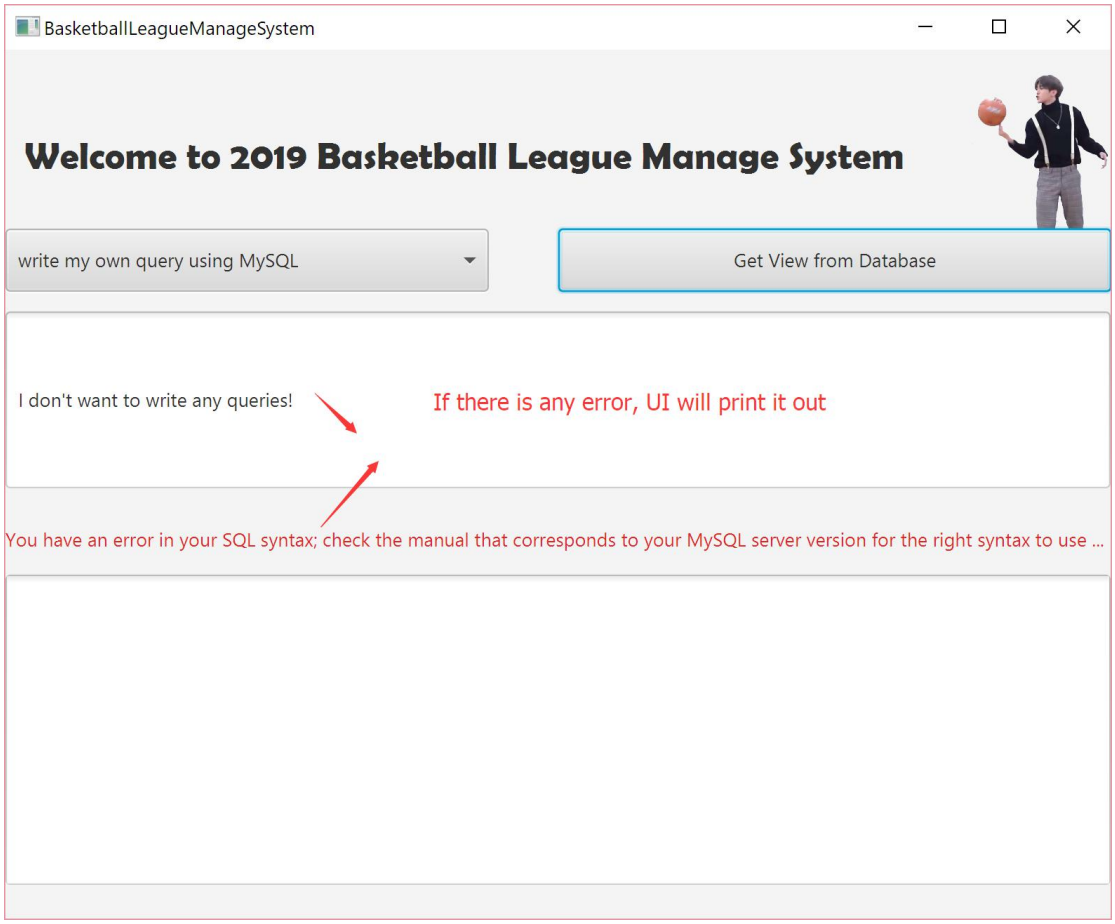


2.4.Get the average age for every team



3. Write user's own queries





4. (Part of) Java codes for JDBC and UI

- *Controller.java*

```
1.  package BMsystem;
2.
3.  import com.mysql.jdbc.exceptions.jdbc4.MySQLSyntaxErrorException;
4.  import javafx.collections.FXCollections;
5.  import javafx.event.ActionEvent;
6.  import javafx.fxml.Initializable;
7.  import javafx.scene.control.*;
8.  import javafx.scene.image.Image;
9.  import javafx.scene.image.ImageView;
10. import java.net.URL;
11. import java.sql.*;
12. import java.util.ArrayList;
13. import java.util.List;
14. import java.util.ResourceBundle;
15.
16. public class Controller implements Initializable {
17.     public TextField sqlText;
```

```
18.     public ChoiceBox cb;
19.     public Label warning;
20.     public TextArea viewList;
21.     public ImageView image;
22.     private Connection conn;
23.
24.     public void getView(ActionEvent actionEvent) throws SQLException {
25.         warning.setText("");
26.          if (cb.getSelectionModel().getSelectedItem().toString().equals("writ
    e my own query using MySQL")) {
27.             String stmt = sqlText.getText();
28.             if (stmt != null && stmt.length() != 0) {
29.                 PreparedStatement p = conn.prepareStatement(stmt);
30.                 executeQuery(p);
31.             } else {
32.                 warning.setText("Please enter some queries!");
33.             }
34.         } else if (cb.getSelectionModel().getSelectedItem() == null || cb.ge
    tSelectionModel().getSelectedItem().toString().length() == 0) {
35.             warning.setText("Please choose your management!");
36.         } else if (cb.getSelectionModel().getSelectedItem().toString().equal
    s("show the score table for this year's points")) {
37.             String stmt =
38.                     "select team_name as Name, (win_number+draw_number+loss_
    number) as Played, win_number as Win,draw_number as Draw, loss_number as Loss,
     (win_number*2+draw_number) as Points " +
39.                             "from Team;";
40.             PreparedStatement p = conn.prepareStatement(stmt);
41.             executeQuery(p);
42.         } else if (cb.getSelectionModel().getSelectedItem().toString().equal
    s("show the score table for this year's and next year's points")) {
43.             String stmt =
44.                     "select team_name as Name, (win_number+draw_number+loss_
    number) as Played, win_number as Win,draw_number as Draw, loss_number as Loss,
     (win_number*2+draw_number) as Points, (win_number*3+draw_number*2-loss_numbe
    r*1) as 'Next Year\\'s Points' " +
45.                             "from Team;";
46.             PreparedStatement p = conn.prepareStatement(stmt);
47.             executeQuery(p);

48. ......

49. ......

50.     @Override
```

```
51.     public void initialize(URL location, ResourceBundle resources) {
52.          Image imageSource = new Image("file:C:\\Users\\gmf\\Desktop\\数据库实
    验\\BasketballLeageManageSystem\\641.png");
53.         image.setImage(imageSource);
54.         cb.setItems(FXCollections.observableArrayList(
55.                 "show the score table for this year's points", "show the sco
    re table for this year's and next year's points", "Look for a match using awa
    y team name, home team name and date",
56.                 "Look for all the matches on one day", "Look for all the mat
    ches between two teams", "Look for attributes of a player by ID",
57.                 "Get the average age for every team","Look for the youngest
    player in a team","Look for players bought by a coach/manager order by prices
     in descending order",
58.                 "Look for a team's players whose age is within a range","Loo
    k for all of a team's matches as home team","Look for all the teams in one ci
    ty",
59.                 "Look for all satisfied players that has Estimated Value wit
    hin a range","Look for a team's manager/coach and his attributes",
60.                 "According to coach name, look for his players buying year &
     price","Look for players that works for a certain team by a team name",
61.                "Look for all players with the same family name",
62.                new Separator(), "write my own query using MySQL")
63.         );
64.         try {
65.             Class.forName("com.mysql.jdbc.Driver");
66.         } catch (ClassNotFoundException e) {
67.             System.out.println("Driver could not be loaded");
68.             System.exit(0);
69.         }
70.         try {
71.              conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/
    " + "basketball_league", "root", "205173");
72.         } catch (SQLException e) {
73.             e.printStackTrace();
74.         }
75.     }
76.
77.     private void executeQuery(PreparedStatement p) {
78.         try {
79.             ResultSet rs = p.executeQuery();
80.             int col = rs.getMetaData().getColumnCount();
81.             StringBuilder stringBuilder = new StringBuilder();
82.             List<List<String>> data = new ArrayList<>();
83.             List<String> temp = new ArrayList<>();
```

```
84.             for (int j = 0; j < rs.getMetaData().getColumnCount(); j++) {
85.                 temp.add(rs.getMetaData().getColumnName(j+1));
86.             }
87.             data.add(temp);
88.             while (rs.next()) {
89.                 temp = new ArrayList<>();
90.                 for (int i = 1; i <= col; i++) {
91.                     temp.add(rs.getString(i) );
92.                 }
93.                 data.add(temp);
94.             }
95.             viewList.setText(new TableFormat(data).println(10));
96.
97.         } catch (MySQLSyntaxErrorException e) {
98.             warning.setText(e.getMessage());
99.         } catch (SQLException e) {
100.            e.printStackTrace();
101.        }
102.    }
103.}
```

- *TableFormat.java*

```
1.  package BMsystem;
2.
3.  import java.util.ArrayList;
4.  import java.util.List;
5.
6.  public class TableFormat {
7.
8.      public List<List<String>> data;
9.
10.     public String println(Integer interval) {
11.         StringBuilder stringBuilder = new StringBuilder();
12.         Integer width = data.get(0).size();
13.         Integer high = data.size();
14.         Integer[] maxWidths = getMaxWidth();
15.         for (int i = 0; i < high; i++) {
16.             for (int y = 0; y < width; y++) {
17.                 String text = data.get(i).get(y);
18.                 Integer maxWidth = maxWidths[y];
19.                 if (y > 0) {
20.                     maxWidth+=interval;
21.                 }
22.                 stringBuilder.append(getPlace(text, maxWidth));
```

```
            }
            stringBuilder.append("\n");
        }
        return stringBuilder.toString();
    }

    public String getPlace(String text, Integer maxWidth) {
        int textSize = text.length();
        for (int i = 0; i < maxWidth - textSize; i++) {
            text = " " + text;
        }
        return text;
    }

    /**
     * calculate the max size of each line
     */
    public Integer[] getMaxWidth() {
        Integer width = data.get(0).size();
        Integer high = data.size();
        Integer[] widthArray = new Integer[width];
        for (int w = 0; w < width; w++) {
            String[] array = new String[high];
            for (int h = 0; h < high; h++) {
                array[h] = data.get(h).get(w);
            }
            widthArray[w] = getLengthMax(array);
        }
        return widthArray;
    }

    /**
     * get the max value of elements in an array
     */
    public Integer getLengthMax(String[] arr) {
        Integer max = arr[0].length();
        for (int i = 1; i < arr.length; i++) {
            if (arr[i].length() > max) {
                max = arr[i].length();
            }
        }
        return max;
    }

.....
```

- *Main.java*

```java
package sample;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class Main extends Application {

    @Override
    public void start(Stage primaryStage) throws Exception{
        Parent root = FXMLLoader.load(getClass().getResource("sample.fxml"));

        primaryStage.setTitle("BasketballLeagueManageSystem");
        primaryStage.setScene(new Scene(root, 700, 550));
        primaryStage.show();
    }


    public static void main(String[] args) {
        launch(args);
    }
}
```