BIG DATA PROGRAMMING HOMEWORK #3 UNDERGRADUATE STUDENT NAME: HAMA EARVIN BAKO

In [ ]:

```
import sqlite3
```

In [ ]:

```
conn = sqlite3.connect('database.sqlite')
```

In [ ]:

```
cursor = conn.cursor()
```

Question 1 (20 points): Write a SQL query that lists all the players born between 1987 and 1990 inclusive, sort them from the oldest to the youngest. The output of this query should be of the form: Player Name | Birthday

In [ ]:

```
cursor.execute("SELECT player_name, birthday FROM PLAYER WHERE birthday >=1987 AND
birthday <= 1990 ORDER BY birthday ASC;")
```

In [ ]:

```
rows = cursor.fetchall()
```

In [ ]:

```
for row in rows:
    print(row)
```

Question 2 (20 points): Write a SQL query that ranks all countries and leagues based on the total amount of total goals scored per game in the whole dataset. Sort them by the largest to the smallest amount of goals. Note: Read this carefully. The output of this query should be of the form: Country | League Name |Total Goals Scored

In [ ]:

```
cursor.execute("SELECT DISTINCT Country.name, League.name, SUM(Match.home_team_goal
+ Match.away_team_goal) FROM Country, League, Match WHERE Country.id =
League.country_id GROUP BY Country.id ORDER BY SUM(Match.home_team_goal +
Match.away_team_goal) ASC;")
```

In [ ]:

```
rows = cursor.fetchall()
```

In [ ]:

```
#The answer is wrong because the data is corrupted
#The country id and league id for the Match table is always 1
for row in rows:
    print(row)
```

Question 3 (20 points): Write a SQL query that ranks all teams by the average of all their attributes (not the players' attributes), sort them from best to worst. The output of this query should be of the form: Team Long Name | Average of Attributes

In [ ]:

```
cursor.execute("SELECT DISTINCT team_long_name, (BuildUpPLaySpeed +
BuildUpPlayDribbling + BuildUpPlayPassing + chanceCreationPassing +
chanceCreationCrossing + chanceCreationShooting + defencePressure +
defenceAggression + defenceTeamWidth)/9 FROM Team, Team_Attributes WHERE
Team.team_api_id = Team_Attributes.team_api_id ORDER BY (BuildUpPLaySpeed +
BuildUpPlayDribbling + BuildUpPlayPassing + chanceCreationPassing +
chanceCreationCrossing + chanceCreationShooting + defencePressure +
defenceAggression + defenceTeamWidth)/9 DESC ;")
```

In [ ]:

```
rows = cursor.fetchall()
```

In [ ]:

```
for row in rows:
    print(row)
```

In [ ]:

```
Question 4 (20 points): Write a SQL query that ranks all teams by the average of
their players'
attributes, sort them by descending order displaying only the top 5. The output of
this query should be
of the form:
Team Name| Number of Players | Player Attribute Average
```

In [ ]:

```
cursor.execute("SELECT team_long_name, Count(player_fifa_api_id),
AVG(overall_rating) FROM Team, PLayer_Attributes WHERE team_fifa_api_id =
player_fifa_api_id GROUP BY team_fifa_api_id ORDER BY AVG(overall_rating) DESC LIMIT
5;")
```

rows = cursor.fetchall()

In [ ]:

```
for row in rows:
    print(row)
```

Question 5 (40 points): Write a SINGLE SQL query that finds the date that had the most goals scored on, per each different season and league. The output of this query should be of the form: Date (dd/mm/yy) | Season | League Name | Goals scored

In [ ]:

```
cursor.execute("SELECT DISTINCT Match.date, Match.season, League.name,
```

```
SUM(Match.home_team_goal + Match.away_team_goal) FROM Match, League WHERE
Match.league_id = League.id GROUP BY Match.date ORDER BY SUM(Match.home_team_goal +
Match.away_team_goal) DESC LIMIT 5;")
```

In [ ]:

```
rows = cursor.fetchall()
```

In [ ]:

```
for row in rows:
    print(row)
```

Graduate Student Task (40 points): Write a SINGLE SQL query that finds the top 5 teams in terms of goals scored PER league for the 2008/2009 season. The output of this query should be of the form:

In [ ]:

```
cursor.execute("SELECT DISTINCT team_long_name, Match.season, League.name,
SUM(Match.home_team_goal + Match.away_team_goal) FROM Team, League, Match WHERE
Match.season LIKE '%2008/2009%' AND Match.League_id = League.id  AND (team_api_id =
home_team_api_id OR team_api_id = away_team_api_id) GROUP BY team_api_id ORDER BY
SUM(Match.home_team_goal + Match.away_team_goal) DESC LIMIT 5;")
```

In [ ]:

```
rows = cursor.fetchall()
```

In [ ]:

```
#I wasn't able to get each league top 5 teams in term of goals scored per league
#Instead I just got the overalll top 5 teams
#The only way I can think of doing it is through UNION and make 9 SELECT statement
for eah league
#So I just stop here. I feel like I am close, but I can't find the way to nest it
properly
for row in rows:
    print(row)
```