# Database Applications Exam (March 2015) – Football

Your exam consists of several parts, explained below. You may work independently on each exam part. Submit your solutions as a single **ZIP file holding the full source code**, without any libraries and compiled binaries.

## Part I – Query Existing Database, Import and Export Data

You are given a **MS SQL Server database "Football"** holding countries, football teams, football leagues, and football matches, available as **SQL script**. Your task is to write a few data-driven applications in C# for importing data, querying data and exporting data from the database.

### Entity Framework Mappings (Database First)

Create an **Entity Framework (EF) data model** of the existing database (map the database tables to C# classes). Use the "**database first**" model in EF. To test your EF data model, **list all team names**.

3 score

Make sure all navigation properties have **good (self-describing) names**.

2 score

### Export the Leagues and Teams as JSON

Write a **C# application** based on your EF data model for **exporting all leagues along with their teams** in the following JSON format:

|  |
| --- |
| **leagues-and-teams.json** |
| [  { "leagueName": "FIFA 2018 World Cup", "teams": [ ] },  { "leagueName": "German Bundesliga", "teams": [ "Bayer Leverkusen", "Bayern Munich", "Borussia Dortmund", "Borussia Monchengladbach", … ] },  { "leagueName": "Italian Serie A", "teams": [ "AC Milan", "AS Roma", … ] },  { "leagueName": "UEFA Champions League", "teams": [ "Arsenal", … ] },  { "leagueName": "UK Premier League", "teams": [ "Arsenal", "Aston Villa", … ] }  ] |

Write the output in a JSON file named **leagues-and-teams.json**. Include in the output the leagues with no teams (if any). The code indentation in the JSON file is not important.

6 score

Order the **leagues** and the **teams in each league alphabetically**.

2 score

For better performance, ensure your program executes a **single DB query** and retrieves from the database only the required data (without unneeded rows and columns).

2 score

### Export International Matches as XML

Write a **C# application** based on your EF data model for **exporting all international matches and their score** in a XML file named international-matches.xml in the following XML format:

|  |
| --- |
| **international-matches.xml** |
| <?xml version="1.0" encoding="utf-8"?>  <matches>  <match>  <home-country code="ES">Spain</home-country>  <away-country code="BG">Bulgaria</away-country>  <league>FIFA 2018 World Cup</league>  </match>  <match>  <home-country code="ES">Spain</home-country>  <away-country code="DE">Germany</away-country>  </match>  <match date-time="26-Jun-1994 11:35">  <home-country code="BG">Bulgaria</home-country>  <away-country code="GR">Greece</away-country>  <score>4-0</score>  </match>  <match date-time="30-Jun-1994 06:35">  <home-country code="BG">Bulgaria</home-country>  <away-country code="AR">Argentina</away-country>  <score>2-0</score>  </match>  <match date="10-Jul-1994">  <home-country code="DE">Germany</home-country>  <away-country code="BG">Bulgaria</away-country>  <score>1-2</score>  </match>  <match date="25-Mar-2015">  <home-country code="DK">Denmark</home-country>  <away-country code="US">United States</away-country>  <score>3-2</score>  </match>  <match date-time="25-Mar-2015 07:00">  <home-country code="GE">Georgia</home-country>  <away-country code="MT">Malta</away-country>  <score>2-0</score>  </match>  <match date-time="13-Jun-2015 09:45">  <home-country code="AL">Albania</home-country>  <away-country code="FR">France</away-country>  <league>FIFA 2018 World Cup</league>  </match>  </matches> |

**Each match** should have **home country** and **away country**. Each **country** should have **country code** (as attribute). Use an XML parser by choice.

4 score

Attach attribute "**date-time**" when the match has date and time. Attach attribute "**date**" when the match has date only (without time). Do not attach any attributes when the match has **no date**.

4 score

List the **score** if the match has score (home goals and away goals). List the **league** name if the match has a league.

3 score

Order the **ma0tches** by **date** (from the earliest) and by **home country** and **away country** alphabetically as second and third criteria.

2 score

For better performance, ensure your program executes a **single DB query** and retrieves from the database only the required data (without unneeded rows and columns).

2 score

### Import Leagues and Teams from XML

Write a **C# application** based on your EF data model for **importing leagues and teams**. The application should process **multiple requests** and write **logs** for each operation at the console.

The input comes from an XML file leagues-and-teams.xml in the following format:

|  |
| --- |
| **leagues-and-teams.xml** |
| <?xml version="1.0" encoding="utf-8" ?>  <leagues-and-teams>  <league>  <league-name>Unofficial Friendly Games</league-name>  <teams>  <team name="Krivokracite" country="Bulgaria" />  <team name="Hayduk" country="Serbia" />  <team name="Amatyor" country="Bulgaria" />  <team name="Unknown Team" />  <team name="Piyani Kashkavali" country="Bulgaria" />  <team name="Dai Hui Tu Chou" country="China" />  <team name="Chelsea" country="United Kingdom" />  <team name="Manchester United" country="United Kingdom" />  <team name="Chelsea" country="Bulgaria" />  </teams>  </league>  <league>  <league-name>UK Premier League</league-name>  <teams>  <team name="Manchester United" country="United Kingdom" />  <team name="Arsenal" country="United Kingdom" />  </teams>  </league>  <league>  <league-name>German Bundesliga</league-name>  </league>  <league>  <league-name>New League</league-name>  </league>  <league>  <teams>  <team name="New Team 1" country="Bulgaria" />  <team name="New Team 2" country="France" />  <team name="New Team 3" />  </teams>  </league>  <league />  </leagues-and-teams> |

The input XML holds a sequence of requests given in the **<league>…</league>** elements.

The element "**league-name**" is optional. The specified league should be created in the database, if it does not exist.

The elements "**teams**" and "**team**" are optional. The specified teams should be created in the database, if they do not exist. Note that team "**name**" is mandatory, but team "**country**" is optional. A team is considered existing in the database when it is matched by both name and country.

If both elements "**league-name**" and "**teams**" exist in a certain query, all teams, not in the given league, should be added to it.

The **output** should be printed on the console in the following format:

|  |
| --- |
| Processing league #1 ...  Created league: Unofficial Friendly Games  Created team: Krivokracite (Bulgaria)  Added team to league: Krivokracite to league Unofficial Friendly Games  Created team: Hayduk (Serbia)  Added team to league: Hayduk to league Unofficial Friendly Games  Created team: Amatyor (Bulgaria)  Added team to league: Amatyor to league Unofficial Friendly Games  Created team: Unknown Team (no country)  Added team to league: Unknown Team to league Unofficial Friendly Games  Created team: Piyani Kashkavali (Bulgaria)  Added team to league: Piyani Kashkavali to league Unofficial Friendly Games  Created team: Dai Hui Tu Chou (China)  Added team to league: Dai Hui Tu Chou to league Unofficial Friendly Games  Existing team: Chelsea (United Kingdom)  Added team to league: Chelsea to league Unofficial Friendly Games  Existing team: Manchester United (United Kingdom)  Added team to league: Manchester United to league Unofficial Friendly Games  Created team: Chelsea (Bulgaria)  Added team to league: Chelsea to league Unofficial Friendly Games  Processing league #2 ...  Existing league: UK Premier League  Existing team: Manchester United (United Kingdom)  Existing team in league: Manchester United belongs to UK Premier League  Existing team: Arsenal (United Kingdom)  Existing team in league: Arsenal belongs to UK Premier League  Processing league #3 ...  Existing league: German Bundesliga  Processing league #4 ...  Created league: New League  Processing league #5 ...  Created team: New Team 1 (Bulgaria)  Created team: New Team 2 (France)  Created team: New Team 3 (no country)  Processing league #6 ... |

The above result should be produced when the program is executed over a **clean database** (as it is initially given in the database create script).

If the program is executed twice, its **second output** should be as follows:

|  |
| --- |
| Processing league #1 ...  Existing league: Unofficial Friendly Games  Existing team: Krivokracite (Bulgaria)  Existing team in league: Krivokracite belongs to Unofficial Friendly Games  Existing team: Hayduk (Serbia)  Existing team in league: Hayduk belongs to Unofficial Friendly Games  Existing team: Amatyor (Bulgaria)  Existing team in league: Amatyor belongs to Unofficial Friendly Games  Existing team: Unknown Team (no country)  Existing team in league: Unknown Team belongs to Unofficial Friendly Games  Existing team: Piyani Kashkavali (Bulgaria)  Existing team in league: Piyani Kashkavali belongs to Unofficial Friendly Games  Existing team: Dai Hui Tu Chou (China)  Existing team in league: Dai Hui Tu Chou belongs to Unofficial Friendly Games  Existing team: Chelsea (United Kingdom)  Existing team in league: Chelsea belongs to Unofficial Friendly Games  Existing team: Manchester United (United Kingdom)  Existing team in league: Manchester United belongs to Unofficial Friendly Games  Existing team: Chelsea (Bulgaria)  Existing team in league: Chelsea belongs to Unofficial Friendly Games  Processing league #2 ...  Existing league: UK Premier League  Existing team: Manchester United (United Kingdom)  Existing team in league: Manchester United belongs to UK Premier League  Existing team: Arsenal (United Kingdom)  Existing team in league: Arsenal belongs to UK Premier League  Processing league #3 ...  Existing league: German Bundesliga  Processing league #4 ...  Existing league: New League  Processing league #5 ...  Existing team: New Team 1 (Bulgaria)  Existing team: New Team 2 (France)  Existing team: New Team 3 (no country)  Processing league #6 ... |

Your program should correctly **parse the input XML**.

5 score

Your program should correctly **import leagues** (new and existing).

7 score

Your program should correctly **import teams** (new and existing).

12 score

Your program should correctly **add teams to leagues** (when a team does not already belong to a league).

6 score

### \* Generate Random Matches

Write a **C# application** based on your EF data model for **generating random matches with random score**. The application should process **multiple requests** and write **logs** for each operation at the console.

The input comes from an XML file generate-matches.xml in the following format:

|  |
| --- |
| **generate-matches.xml** |
| <?xml version="1.0" encoding="utf-8" ?>  <generate-random-matches>  <generate generate-count="2" max-goals="4">  <league>UK Premier League</league>  <start-date>1-Jan-2015</start-date>  <end-date>20-Jun-2015</end-date>  </generate>  <generate />  <generate generate-count="3" max-goals="10">  <start-date>10-Jan-2000</start-date>  <end-date>20-Sep-2002</end-date>  </generate>  <generate generate-count="5">  <league>German Bundesliga</league>  </generate>  </generate-random-matches> |

All elements and attributes in the input XML are **non-mandatory**.

* When "**generate-count**" is not specified, it has default value of **10**.
* When "**max-goals**" is not specified, it has default value of **5**.
* When "**start-date**" is not specified, it has default value of "**1-Jan-2000**".
* When "**end-date**" is not specified, it has default value of "**31-Dec-2015**".

The output should be printed at the console in the following format:

|  |
| --- |
| Processing request #1 ...  5-Feb-2015: Everton - West Ham United: 1-2 (UK Premier League)  14-Mar-2015: Hull City - Liverpool: 4-3 (UK Premier League)  Processing request #2 ...  08-Dec-2002: Juventus - Atalanta: 0-5 (no league)  02-Nov-2012: Eintracht Frankfurt - Udinese: 2-4 (no league)  25-Dec-2015: Real Madrid - Sunderland: 3-4 (no league)  11-Jan-2000: Hellas Verona - Shakhtar Donetsk: 5-2 (no league)  29-Oct-2007: Udinese - Atalanta: 3-2 (no league)  28-Mar-2013: Porto - Porto: 0-2 (no league)  25-Apr-2011: Bayern Munich - Lazio: 2-0 (no league)  24-Feb-2003: VfB Stuttgart - Paris Saint-Germain: 0-1 (no league)  21-Mar-2015: Everton - Porto: 5-4 (no league)  16-Feb-2004: Udinese - Bayer Leverkusen: 4-0 (no league)  Processing request #3 ...  24-Oct-2000: Queens Park Rangers - Borussia Dortmund: 7-6 (no league)  05-Mar-2002: Mainz - AS Monaco FC: 9-9 (no league)  21-Feb-2001: Genoa - Southampton: 10-7 (no league)  Processing request #4 ...  22-Jul-2007: Mainz - Eintracht Frankfurt: 5-5 (German Bundesliga)  19-Oct-2019: Bayer Leverkusen - SC Paderborn 07: 5-3 (German Bundesliga)  07-Nov-2009: Bayer Leverkusen - VfL Wolfsburg: 5-0 (German Bundesliga)  13-Jun-2010: FC Cologne - VfL Wolfsburg: 0-4 (German Bundesliga)  23-Jul-2020: Hannover 96 - SC Freiburg: 3-4 (German Bundesliga) |

Your program should correctly **parse the input XML**.

4 score (bonus)

Your program should correctly **generate random matches**.

12 score (bonus)

Your program should correctly **save in the database** the generated random matches.

2 score (bonus)

Your program should correctly **print to the console** the generated random matches.

2 score (bonus)

## Part II – EF Code First: Define Data Model, Import and Export Data

You are assigned to define a **code first data model in EF** and write a few data-driven applications in C# for importing data, querying data and exporting data from the database.

Use a new database "**Phonebook**". Do not modify the "**Football**" database.

### EF Code First: Phonebook

Create an **Entity Framework (EF) code first data model** for keeping phonebook holding contacts with phones and emails. It should have several entities:

* **Contacts** have **name** and optionally **position**, **company**, **emails**, **phones**, **site** (URL) and **notes** (free text).
* **Emails** hold **email address**.
* **Phones** hold **phone number**.

7 score

**Seed** your database with a few **contacts**, using the **EF migrations** framework. It is OK to drop the database in case of model changes or use any other migration strategy like automatic upgrade to the latest DB schema. Insert the following **contacts** in your seed method:

|  |
| --- |
| Name: Peter Ivanov  Position: CTO  Company: Smart Ideas  Emails: peter@gmail.com, peter\_ivanov@yahoo.com  Phones: +359 2 22 22 22, +359 88 77 88 99  Url: http://blog.peter.com  Notes: Friend from school  Name: Maria  Phone: +359 22 33 44 55  Name: Angie Stanton  Email: info@angiestanton.com  Url: http://angiestanton.com |

5 score

To test your data model, **list all contacts along with their phones and emails**.

3 score

### Import Contacts from JSON

Write a **C# application** based on your EF code first data model for **importing into the DB a set of phonebook contacts** given in the file contacts.json in the following JSON format:

|  |
| --- |
| **contacts.json** |
| [  { "name" : "Steve", "phones" : [ "088 1234 5678" ] },  { "name" : "Gabriela", "emails" : [ "gaby@yahoo.com", "gaby.d@gmail.com" ] },  { },  { "name" : "Milena Smilyanova",  "company" : "Mente Soft Ltd.",  "position" : "CEO",  "site" : "http://mentesoft.org",  "phones" : [ "088 77 22 77", "02 / 981 77 22" ],  "emails" : [ "gaby@yahoo.com", "gaby.d@gmail.com" ],  "site" : "http://mentesoft.org",  "notes" : "Friends from Plovdiv International Fair" },  { "firstName" : "Steve", "lastName" : "Balmer", "email" : "steve@ms.org" },  { "name" : "Unknown Contact" }  ] |

The only mandatory property is **name**. All the others are **optional**.

You should **parse the JSON** and throw an exception in case of incorrect data, e.g. when a required element is missing or an invalid value is given. The size of the JSON file will be less than **10 MB**. Use a JSON parser by choice.

10 score

You should correctly **import the contacts into the DB**.

5 score

You should correctly import the **emails and phones** into the DB.

5 score

Insert each contact in a **separate transaction**. A contact should either be inserted correctly along with all its emails and phones, or no part of it should be inserted at all. Print as **output** a single line for each contact from the input JSON: either "*Contact <name> imported*" or "*Error: <error message>*". Error messages should describe briefly the problem (as free text) and may optionally include exception stack-trace. Sample output:

|  |
| --- |
| Contact Steve imported  Contact Gabriela imported  Error: Name is required  Contact Milena Smilyanova imported  Error: Name is required  Contact Unknown Contact imported |

5 score

## Exam Information

You are allowed to use any resources you have, e.g. Internet, software, existing code.

You are not allowed to get help from other people. Skype, ICQ, FB, email, talks, phone calls, etc. are forbidden.

Exam time: **6 hours**.