# Database Systems – Sample Exam – July 2013

You are assigned to design data model and data access logic for an online **bookmarking site**. The site should hold **users** and each user should have a set of **bookmarks**. Each bookmark has a title, an URL, and optional notes. Bookmarks can be tagged with a set of **tags**. Users have username, optional password (MD5 hash) and optional name. Tags are words or phrases, e.g. "SQL" or "Entity Framework".

### Bookmarks Database

Design a database schema in SQL Server to keep the **users**, **bookmarks** and **tags**.

Provide a **SQL script for your database schema**. Script the schema only without any data in it.

10 score

### Database Tuning

Ensure you have defined appropriate **constraints** to ensure data integrity. Ensure you have defined appropriate database **indexes** to allow fast searching by username, by title, and by set of tags. Fast case-insensitive searching by any word contained in the bookmark's title should be supported.

\* The easiest way to implement searching in the bookmark titles is by automatically tag the bookmark with all words contained in its title.

\* You are not allowed to use "Full-Text Search" in MS SQL Server.

5 score (optional)

### Import Bookmarks from XML File

Write a **C# program** for **importing into the DB a set of bookmarks** given in XML file "**bookmarks.xml**":

|  |
| --- |
| **bookmarks.xml** |
| <?xml version="1.0" ?>  <bookmarks>  <bookmark>  <username>peter</username>  <title>Nakov's books</title>  <url>http://www.nakov.com/books/</url>  <tags>nakov, programming, book</tags>  </bookmark>  <bookmark>  <username>todor</username>  <title>HTML 5 Demos and Examples</title>  <url>http://html5demos.com</url>  <tags>HTML, HTML5, CSS, CSS3, Web development, web</tags>  </bookmark>  <bookmark>  <title>Telerik Academy</title>  <url>http://academy.telerik.com</url>  <username>maria</username>  </bookmark>  <bookmark>  <username>peter</username>  <title>Intro C# Book</title>  <notes>Fundamental book of programming. A must read!</notes>  <url>http://www.introprogramming.info/intro-csharp-book/</url>  <tags>programming, C#, algorithms, data-structures, nakov, book</tags>  </bookmark>  <bookmark>  <tags>SQL, dummy</tags>  <username>maria</username>  <title>SQL for Dummies</title>  <url>http://www.amazon.com/SQL-For-Dummies-Allen-Taylor/dp/0470557419</url>  </bookmark>  <bookmark>  <url>http://forums.academy.telerik.com</url>  <title>Telerik Academy Forums</title>  <notes>The official forum of Telerik Academy: ask your questions and you will be answered soon.</notes>  <username>peter</username>  </bookmark>  </bookmarks> |

You should **parse the XML** and throw an exception in case of required element is missing. Note that tags and notes are optional while username, title and URL are obligatory. The size of the XML file will be less than **10 MB**.

\* You are free to use an XML parser by choice.

6 score

You should correctly **import the bookmarks, users and tags into the DB**. Keep in mind that the users and the tags are unique. If some user or tag already exists in the database, it should be reused (no duplicates are allowed). The same URL can be bookmarked by multiple users. Tags are given as a sequence of words or phrases separated by commas. Ensure that all words from the bookmark's title are added as tags for this bookmark as part of the import process.

\* You are free to use an ORM framework by choice (in code first or database first approach) or plain ADO.NET.

17 score

Inserting a set of bookmarks should either be **completed entirely or nothing should be changed** in the database. It is not acceptable to insert a piece of the specified XML and skip the rest of it due to errors.

2 score (optional)

### Simple Bookmark Search

Implement a **C# program** for **searching for bookmarks** by **username** and **tag** specified in a XML file "**simple-query.xml**" in the following format:

|  |
| --- |
| **simple-query.xml** |
| <?xml version="1.0" ?>  <query>  <username>peter</username>  <tag>programming</tag>  </query> |

The **username is optional**, but the **tag is mandatory**.

The result should be printed on the console as a list of URLs in **alphabetical order**:

|  |
| --- |
| http://www.introprogramming.info/intro-csharp-book/  http://www.nakov.com/books/ |

In case of nothing found, print "**Nothing found**" at the console.

\* You are free to use an ORM framework by choice (in code first or database first approach) or plain ADO.NET.

\* You are free to an XML parser by choice.

10 score

### Complex Bookmark Search

Implement a **C# program** for **searching for bookmarks** by **username** and a **set of tags**. The program should take as input a sequence of queries from the XML file "**complex-query.xml**" in the format shown below:

|  |
| --- |
| **complex-query.xml** |
| <?xml version="1.0" ?>  <queries>  <query max-results="3">  <username>peter</username>  <tag>programming</tag>  <tag>book</tag>  </query>  <query max-results="1">  <tag>programming</tag>  <tag>nakov</tag>  </query>  <query max-results="100">  <tag>mente</tag>  <username>kaspichan</username>  </query>  <query max-results="1">  <username>maria</username>  </query>  <query max-results="3" />  </queries> |

The **username is optional**. The **set of tags** is arbitrary set of words (empty set is also allowed). The search has **"and" semantics**. This means that searching by tags "programming" and "book" and username "peter" should return all bookmarks from "peter" matching in the same time both tags "programming" and "book". If the username is missing, then the bookmarks from all users should be searched. If the tags are missing, then the bookmark tags should be ignored in the search. If both username and tags are missing in a query, it should return all the bookmarks from the database. The "**max-results**" attribute is optional and specifies how many results to return at most.

Write the results in the XML file "**search-results.xml**" in the following format:

|  |
| --- |
| **search-results.xml** |
| <?xml version="1.0" ?>  <search-results>  <result-set>  <bookmark>  <username>peter</username>  <title>Intro C# Book</title>  <url>http://www.introprogramming.info/intro-csharp-book/</url>  <tags>algorithms, book, c#, data-structures, nakov, programming</tags>  <notes>Fundamental book of programming. A must read!</notes>  </bookmark>  <bookmark>  <username>peter</username>  <title>Nakov's books</title>  <url>http://www.nakov.com/books/</url>  <tags>book, nakov, programming</tags>  </bookmark>  </result-set>  <result-set>  <bookmark>  <username>peter</username>  <title>Intro C# Book</title>  <url>http://www.introprogramming.info/intro-csharp-book/</url>  <tags>algorithms, book, c#, data-structures, nakov, programming</tags>  <notes>Fundamental book of programming. A must read!</notes>  </bookmark>  </result-set>  <result-set>  </result-set>  <result-set>  <bookmark>  <username>maria</username>  <title>Telerik Academy</title>  <url>http://academy.telerik.com</url>  </bookmark>  </result-set>  <result-set>  <bookmark>  <username>maria</username>  <title>Telerik Academy</title>  <url>http://academy.telerik.com</url>  </bookmark>  <bookmark>  <username>maria</username>  <title>SQL for Dummies</title>  <url> http://www.amazon.com/SQL-For-Dummies-Allen-Taylor/dp/0470557419</url>  <tags>dummy, sql</tags>  </bookmark>  <bookmark>  <username>todor</username>  <title>HTML 5 Demos and Examples</title>  <url>http://html5demos.com</url>  <tags>css, css3, html, html5, web, web development</tags>  </bookmark>  </result-set>  </search-results> |

Order the results in each result set in **alphabetical order** by their URL. Return no more than the specified **max-results** number of bookmarks for the corresponding query. If the "**max-results**" attribute is missing, return no more than **10** results.

Display the bookmark details in the following order: username, title, URL, tags, notes. If some of these fields is missing or is empty, skip it in the XML. Display the **tags** for each bookmark in **lowercase**, **ordered alphabetically**, and separated by comma and space.

**Implement the search** functionality correctly. Ensure it works fast for millions of records in the DB, and SQL injection is not possible.

The size of the **input XML** file will be less than **1 MB**.

The size of the **output XML** file will be less than **1 GB**.

\* You are free to use an ORM framework by choice (in code first or database first approach) or plain ADO.NET.

\* You are free to use XML parsers by choice, but take into account the performance.

35 score (optional)

### Bulk Data Generator

Write a C# program to generate at least 30 000 users who own 50 000 bookmarks, tagged by 10 000 tags (each bookmark tagged by up to 5 tags).

\* A good way to implement the bulk data generator is: (1) generate random users, (2) generate random bookmarks (each attached to a random user), (3) generate random tags, (4) attach a random number of existing tags to each bookmark, (5) attach each word from the bookmark's title as tag to it.

\* You are free to use a stored procedure, a sequence of native SQL inserts or inserts through an ORM framework by choice.

\* Your generation process should take less than 5 minutes on average fast machine.

15 score (optional)

## Evaluation Criteria

The evaluation criteria are as follows:

* Correct and complete fulfillment of the requirements.
* Good technical design and appropriate use of technologies.
* High-quality programming code – correctness, readability, maintainability.
* Performance – highly-efficient code.

To pass the exam you need 60 score (of 100 scores total).

## Other Terms

During the exam it is allowed to use any teaching materials, lectures, books, existing source code, and other paper or Internet resources.

Direct or indirect communication with anybody in class or outside is forbidden. This includes but does not limit to technical conversations with other students, using mobile phones, chat software (Skype, ICQ, etc.), email, forum posts, folder synchronization software (like Dropbox), etc.

## Exam Duration

Students are allowed to work up to 8 hours.