

# CSE 532: Project 3

Spring 2006

Due: April 24

## 1 Description

In this project, you are to design and implement the same database application as in Projects 1 and 2, but this time using the eXist DBMS, which supports the XQuery language. If you would like to install eXist on your machine, it is available from <http://exist.sourceforge.net/>. You should download and install eXist under your computer account.

You will create your data using a text editor in plain XML files and then store them in eXist. You do not need to create any front ends for this project. All that is needed is to provide the queries and the XML document against which the queries will be tested on command line.

In addition, you are to design XML Schema document appropriate for this application and include it with your deliverables. Note: With XML you can use full object-oriented design — there can be no excuses due to the limitations of the underlying system (such as Oracle’s limited support for objects.) Your XML data must conform to the XML Schema and you must validate it using one of the available validators, such as:

<http://www.xmlspy.com/>

<http://xml.apache.org/xerces-j/>

Make as many of the constraints as possible be part of your XML schema documents. The rest should be expressed as XQuery queries so that the constraint will be considered violated if the corresponding query returns a non-empty answer. (This approach is analogous to assertions in SQL).

## 2 Queries

Implement and test the following queries taken from Projects 1 and 2:

1. List all clients whom Joe Public was dating between March 7, 2006 and June 22, 2006. Use XML data types to represent dates.
2. List all clients whom Joe Public was dating as of March 7, 2005 and such that they had at least one common feature with that person.

Here, dating a certain person as of, say, March 7, 2005 (or as of some other day) means that dating started at time prior or equal to March 7, 2005 and has not stopped on March 7, 2005 or before.

3. Same as above, but now we want Joe Public’s dates to satisfy at least 5 preferences. (You must use aggregates, so that the query could be easily adapted to a different number of common features.)
4. Find all potential “perfect matches” for Joe Public.

A perfect match for a person,  $p$ , is a DSSS client such that the set of all  $p$ ’s preferences is included in the set of that client’s features.

This query involves a non-trivial use of negation.

5. Find Joe Public’s *indirect* dates as of March 7, 2005.

An indirect date at time T, is either a person whom Joe was dating directly at T; or a person who was dating another person,  $p$ , at T and  $p$  was dating Joe directly at that time; or a person who was dating  $p_1$  at T, who was dating  $p_2$ , who was dating ..., etc., who was dating Joe.

Note that this is a recursive query.

6. For each DSSS client who is over 30 years old, find the average duration of the dates (time between when dating started and the time by which it was over) as of the current date.
7. For each age category of DSSS clients, find the average duration of the dates. Age categories include people of ages 20-29, 30-39, 40-49, etc.

The query must *not* depend on the number of age categories and it must be a single query (which may use XQuery functions as views).

### 3 Documentation and Submission Instructions

All the materials must be burned on a CD and submitted at the end of the class on April 24. The CD must have a clearly identifiable label on it, which should contain:

1. Authors' names
2. CSE 532, Project 3

Note that the label must be on the CD itself: an unlabeled CD in a labeled case is not acceptable, because such a CD can get misplaced.

The CD must include the following documents:

1. **README.txt**. This must contain the exact instructions for the TA explaining how to load your data into the eXist database and how to run the queries. All commands must be given in README.txt file in an easy to read form, which the TA should be able to test by just cutting and pasting. The queries to be run are supposed to be in the file *Queries.xquery*, as explained below.
2. **The design document**. This document must include the following:
  - The Entity-Relationship (ER) design diagram of the complete project.
  - A clear description of XML Schema for the database, including a discussion of your design decisions. Remember that your design should be object-oriented. The actual schema should be in the file *Schema.xsd*.
  - Description of **integrity constraints**, including referential integrity and key constraints. Any constraint that cannot be implemented as a **key** or **keyref** statement and instead is implemented as a query (see an earlier remark) must be included in the file *Queries.xquery*.
  - For each query listed in Section 2, the document must supply a **properly formatted** XQuery formulation of the query. (Note: this data must also be included in the file *Queries.xquery*).
3. A separate document called *Schema.xsd* that contains a clearly formatted XML schema for the database.
4. A separate (clearly formatted) file called *Database.xml*, which will contain all the data.
5. A separate document called *Queries.xquery*, which should list all the queries and non-key/keyref constraints.

All documents must include the following statement at the top:

**I (or We, *if working with a partner*) pledge my (or our) honor that all parts of this project were done by me (or us) alone and without collaboration with anybody else.**

## 4 Teaming

You can choose one partner to do the project **jointly**. You can also choose to work alone, but this will not reduce your scope of work.

There are pitfalls in working with a partner whom you do not know well: it can be a frustrating experience to find out that your partner is piggy-backing on you. It can also happen that one of the partners gets involved in academic misconduct, and this may reflect on you. So, choose your partner carefully!

**The division of work** between partners must be **vertical**, not horizontal. This means that the two partners must collaborate on each part of the project and be on top of what the other partner is doing. Each student must be aware of and understand the techniques used by his/her partner. **Your final document must clearly state who did what part of the job.**