COMP4711 Lab 3 (Winter 2009)
Dates: Jan 26-30, 2009

XML with DTD

Background

Our webapps will need material to work with, and we want to store that using XML. Being prudent IT developers, we want to ensure that the data is valid, so we will also want a DTD file to constrain any XML documents that others might create to provide additional data. Time to practice these :)

This is an individual lab.

Lab Tasks

1. We talked about XML in class ... the W3Schools Basic XML tutorial will walk you through the details.
2. We talked about DTDs in class ... the W3Schools DTD tutorial will walk you through the details.
3. A golf handicap database design is presented for your pleasure, at the end of this document.
4. Devise a suitable set of XML documents for the Player and Course data, suitably constrained by DTDs. Provide 3 or 4 courses and 8 players, each of which has played two games on different courses.
5. You can test your XML document by opening it inside a browser, or by right-clicking and "validating" inside NetBeans
6. You could use an external DTD validator, although the DTD has to be internal to the XML document, and that has caused issues before.
7. Zip up your XML documents and associated (external) DTDs, and submit the zip file to share-in, using a suitable name, like ParryJimComp4711Lab3.zip (note: surname followed by first name).

Marking Guideline

This lab will be marked out of 10 ...

- 3 marks for reasonable DTDs
- 3 marks for well-formed XML documents
- 3 marks for valid XML documents
- 1 marks for complete data

Due by the end of the weekend (sunday 11:59pm)

Golf Clubs and Membership Handicaps Scenario

The following data model is designed to hold information relating Members of a Golf Club and hold information relating to Members, Courses, Golf Rounds and Handicaps. For this scenario we need to define the following facts:

These facts define the requirements which the Database must meet and should be agreed between the Database User and the Database Designer prior to physical creation.

The draft facts have been defined as:

The Entities required should include:

- **Members**
- **Courses**
- Course Hole Details (Information relating to a Hole at a Golf Course)
- Golf Rounds (Rounds played by Members)
- Player Handicaps

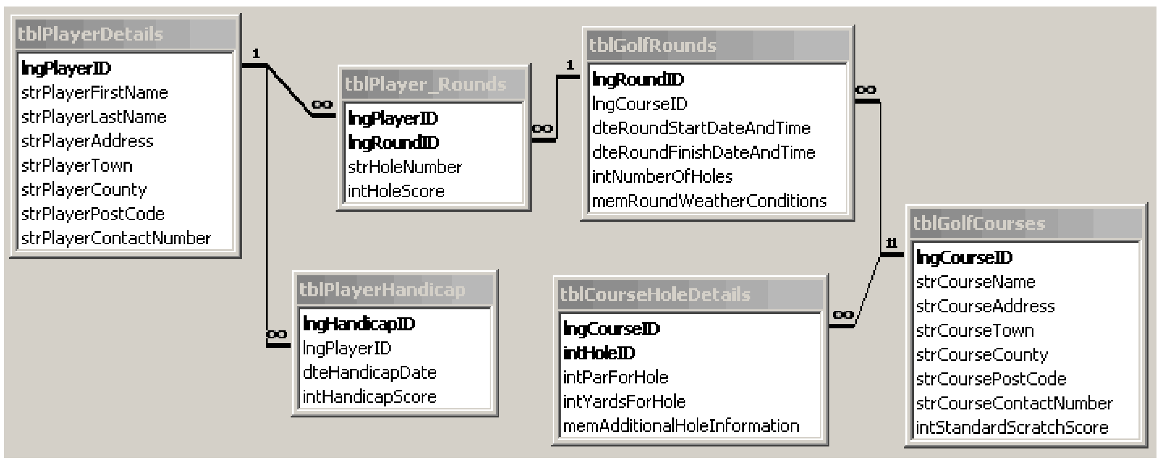
The Entities are related as follows:

- One Member (Player) can play Many Golf Rounds
- One Player can have Many Handicap Reports (Date Dependant)
- One Golf Course can have Many Golf Holes (including Yardage and Par information)
- One Golf Course can provide Many Golf Rounds

When asking questions of the database we may need to know:

1. How many Members (Golf Players) belong to the Golf Club?
2. What score did a particular Member make when playing at a particular Golf Course?
3. What is the Players current Golf Handicap?

The following data model allows these questions to be answered and allows the information contained above to be stored logically and in a structured manner.



Hints

1. You don't need to go overboard with your data ... 9 hole courses are just fine.
2. XML is all about containment, while relational databases are about like tuples stored in individual tables. One way to model the scenario's RDB model in XML would be
    1. Players (PlayerDetails)
        1. Rounds (GolfRounds)
            1. hole Scores (Player_Rounds)
        2. Handicap results (PlayerHandicap)
    2. Courses (GolfCourses)
        1. Hole information (CourseHoleDetails)
3. Players & courses should have unique identifiers, according to the RDB model, and that makes sense. The other “tables” have unique ids because RDB tables require primary keys. Given that we are working with containment, you might be able to avoid some of that primary/foreign key/reference business.
4. No, XML & DTDs on their own cannot answer all of the questions in the scenario, but upcoming tools (schemas and stylesheets) will be able to.
5. Suggestion: use a NetBeans project (a Java web app) to hold your work, and put the XML and DTD documents in the root of the “web pages” tree. You can then “serve” the documents easily, or just validate and preview them from inside the comfort of NB.
6. Almost forgot – I agreed to relax the amount of data required, to 3-4 courses, 3-4 players, and 2 (or more) games recorded (across all the players, not for each).
7. You do not have to use a NetBeans project for this lab, but I think you will find it easier to manipulate and test the XML documents if you do, and you will be better set for the labs two weeks from now, where we will want to be serving our manipulated XML content. If you do use a NB project for this lab, then you can zip up just the XML & DTD files, or the entire project, for your submission.