Cst556 Distributed Applications for .NET with Mono

8. Database Connectivity with .NET
8.a Overview and Motivation

8.a.1 Relational Databases, Odbc and Jdbc

- Storing data in tables: Mapping to C# Objects in the Application

- Open database connectivity (ODBC) for program access to relational data.
- Java database connectivity (JDBC) provides access for Java VM’s.
- ADO.NET provides SQL access to relational databases for .NET languages.
8.a.2 Mono Adaptor

- In Cst556, we will be using the Postgres database that is available for free on a variety of platforms including Unix and Windows.
  - See: http://www.postgresql.org/

- The mono project has produced an .NET adapter for Postgres
  - http://www.mono-project.com/PostgreSQL
8.a.3 Some Structured Query Language (SQL)

- **Insert**

  INSERT INTO Course (CourseID, ProfessorID, Name, Location) 
  SELECT 3, 1, 'Jini and JavaSpaces', 'psa-118';

  In code, the statement: 
  ```java
  String ins = "INSERT INTO registrations VALUES " +
                 "("+i.ToString()+"","+nameTB.Text.Trim()+"","+
                 pathTB.Text.Trim()+"","+organizationTB.Text.Trim()+
                 ","+emailTB.Text.Trim()+","+homeURLTB.Text.Trim()+")";;
  ```

  May result in the following Insert SQL statement:
  ```sql
  INSERT INTO registrations VALUES ('10','airconditioning needs','/cst556/btuCalculator',
                                   'Cst427 ASU at the Polytechnic campus','Tim@asu.edu','http://pooh.azpoly.edu/Cst427');
  ```

- **Update**

  UPDATE Course SET Name = 'Distributed Object Computing', Location = 'cc-120' WHERE CourseID=3;

- **Delete**

  DELETE * FROM Course WHERE CourseID=3;
8.a.4 Some Structured Query Language (SQL)

• **Select**

SELECT Student.Name AS Student, Course.Name AS Course FROM Course, Student, Takes
WHERE Student.StudentID=Takes.StudentID and Course.CourseID=Takes.CourseID
ORDER BY Student.Name;
In code, the statement: string selectSql = "SELECT registryid FROM registrations "+
"WHERE servicepath = " + pathTB.Text.Trim() + "";
May result in the following Select statement:
SELECT registryid FROM registrations WHERE servicepath = '/cst556/demo';

• **Create Table**

CREATE TABLE Student (StudentID INTEGER, Name VARCHAR(32),
CreditsTaken INTEGER)
- The data types available are the defined SQL types
8.a.5  C#'s Object Model versus a Relational Model

- The **mapping** between tables and classes can be problematic
  - Objects as instance attributes of a class
  - Relationships among objects
- **Encapsulation** of implementation details within an object
  - If preserved, performance becomes an issue, for example requiring multiple database connections to satisfy an object
  - Sharing connections causes difficult to manage interactions between C# threads and database locking mechanisms
- What is the **granularity** at which object persistence is managed?
  - Are all tables initially loaded into objects? Or, only when accessed?
  - Caching tables into objects minimizes the number of connections
  - How are modifications in the object reflected in the relational database?
8.b  Using a Database from a C# Application

8.b.1  Making a Connection and Executing a SQL Command

• ADO.NET is a set of classes providing services to .NET programs for accessing relational databases..

• Making a connection and executing an SQL command:

```csharp
private IDataReader reader;
private IDbConnection dbcon;
private IDbCommand dbcmd;
string connectionString =
    "Server=localhost;" +
    "Database=sailboatdb;" +
    "User ID=yourWindowsUserId;" +
    "Password=yourWindowsPassword;"

dbcon = new NpgsqlConnection(connectionString);
dbcon.Open();
dbcmd = dbcon.CreateCommand();
dbcmd.CommandText = "SELECT * FROM boatinformation WHERE model = "
    +"'35' and manufacturer = 'Island Packet';"
reader = dbcmd.ExecuteReader();
```

• Comments

- In this case, the connection class is provided by the Npgsql package
8.b.2  Accessing the Information Read

- One approach is to iterate through the reader picking out values

```csharp
if(reader.Read()) {
    // get the boat dimensions and capacities from the reader
    Double lw1 = (Double)reader["lengthonwater"];  // get the boat dimensions and capacities from the reader
    Double beam = (Double)reader["beam"];
    Double disp = (Double)reader["displacement"];
    Double sa = (Double)reader["sailarea"];
    Double loa = (Double)reader["lengthoverall"];
    String modelYear = (string)reader["year"];
}
```

- For an example, see: sailboatInfoWeb.jar

  - After downloading and extracting the archive, see the file queries.txt for instructions on installing necessary software and running the example.