QT Eğitimi - 5 : Veritabanı - I

Önder Arslan

<onder@xcoders.net>

Sürüm 1.0

Özet

Bu belge qt öğrenimi için bir dizi şeklinde hazırlanmıştır. C++ ve sql bilmek ön koşuldur.

Giriş

Program yazarken en keyif aldığım konulardan birisi veritabanı programlamadır. Sanırım bunun en önemli sebebi veritabanı yönetim sistemleri (RDBMS) ile uzun süre çalışmış olmam (linux üzerinde olmasa da). Uzun zamandır küçük bir telefon defteri uygulaması için bile veritabanı kullanırım. Sıralı dosyalar, rastgele erişimli dosyalar vs. 'ye hiç bir zaman ısınamamışımdır. Bu yüzden kullandığım programlama araçlarının veritabanlarına verdiği destek benim için ayrı bir önem taşıyor. Yani demek istiyorum ki eğitim - 5 ve iki satır sql kodu yazmanın zamanı geldi ;)

insert, update, delete, select

Kodlamaya geçmeden önce bilgisayarınız da MySQL server 'ın kurulu olduğunu, ansi-SQL bildiğinizi yada en azından insert, update, delete işlemleri yapan sql kodlarını yazabildiğinizi varsayıyorum. İlk olarak yapılması gereken Mysql server üzerinde bir veritabanı oluşturmak, ben veritabanı ismi olarak **test** kullanacağım. Verileri silip, değiştirip, güncelleyeceğim tablomun adı da **tablo** olacak. Son olarak tabloma iki alan ekliyorum bunlardan birincisi int türde veri tutan ve değeri otomatik olarak artan (auto_increment) **alan1**, ikincisi varchar türde veri tutan **alan2**.

```
#include <QApplication>
#include <QtSql>
#include <iostream>


using namespace std;

bool connect_data()
{
	QSqlDatabase db = QSqlDatabase::addDatabase("QMYSQL");
	db.setHostName("localhost");
	db.setDatabaseName("test");
	db.setUserName("kullanici_adi");
	db.setPassword("parola");
	bool ok ;
	return ok= db.open();
}


int main(int argc, char *argv[])
{
   QApplication app(argc, argv);
```

```
    if (connect_data())
    {

        QSqlQuery insert_query;
        insert_query.exec("insert into tablo (alan2) values ('ondel')");

        QSqlQuery update_query;
        update_query.exec("update tablo set alan2='onder' where
alan2='ondel'");

        QSqlQuery query;
        query.exec("SELECT alan1,alan2 from tablo");

        QSqlQuery delete_query;
        delete_query.exec("delete from tablo where alan2='onder'");


        while (query.next())
        {
            int alan1 = query.value(0).toInt();
            string alan2 = query.value(1).toString().toStdString();
            cout << alan1 << " "<< alan2 << endl;
        }

    }

    return 0;

}
```

test.cpp:

Qt ile veritabanına bağlanırken yapmamız gereken ilk şey **QtSql** modülünü kodumuza dahil etmek. QtSql modülü qt uygulamalarımız içi kusursuz veritabanı entegrasyonu sağlıyor. QtSql üç katmandan oluşuyor : Driver Layer, SQL API layer ve User Interface layer. Kısaca :
Driver Layer(Sürücü Katmanı) SQL API Layer ile belirli veritabanları arasında köprü görevi görüyor.
SQL API Layer 'ı veritabanına erişmek ve veriler üzerinde çeşitli sorgular çalıştırmak için kullanıyoruz.
User Interface Layer (Kullanıcı arayüz katmanı) verilerin qt kontrolleri üzerinde direk gösterilebilmesi için bağlantı oluşturmaya yarıyor.
QtSql classlarının ayrıntılarını ilerleyen derslerde inceleyeceğiz.

Veritabanına uygulamalarında yapmamız gereken ilk şey tabii ki veritabanına bağlanmak :)) Bunun için programa ilk olarak connect_data() fonksiyonunu yazarak başladım. Bu fonksiyon içinde ilk olarak **QSqlDatabase** türünden db adlı bir nesne oluşturuyorum ve buna **addDatabase** fonksiyonu ile sürücü tipi mysql olacak şekilde ilk değer veriyorum. Qt aşağıda verilen popüler veritabanlarına direk bağlanabilmeniz için driverlara sahiptir. Bunların dışında bir veritabanı kullanacaksanız **registerSqlDriver()** fonksiyonu ile driverı uygulamanıza ekleyebilirsiniz. Ben mysql kulandığım için **QMYSQL** tipini kullandım.

Driver Type	Description
QDB2	IBM DB2
QIBASE	Borland InterBase Driver
QMYSQL	MySQL Driver

Driver Type	Description
QOCI	Oracle Call Interface Driver
QODBC	ODBC Driver (includes Microsoft SQL Server)
QPSQL	PostgreSQL Driver
QSQLITE	SQLite version 3 or above
QSQLITE2	SQLite version 2
QTDS	Sybase Adaptive Server

Daha sonra oluşturduğum db nesnesine setHostName, setDatabase, setUserName, setPassword fonksiyonları ile uygun değerleri atıyorum. Nesnemin open fonksiyonu ile bağlantıyı açıyorum ve bunu geri dönüş değeri (true / false) olarak döndürüyorum. QtSql ile veri tabanı bağlantısı oluşturmak bu kadar kolay ;)

main fonksiyonu içnde ilk olarak app adında QApplication nesnesi oluşturuyoruz. Bu satır qt ile veritabanı uygulaması geliştirmeye çalışırken en çok hata yapmama neden olan satırdır. Bağlantı kodlarımı test ederken bu satırı yazmamam sebebiyle günlerce mysql driver 'ı ile problemim olduğunu düşündüm. Çünkü eğer bu satırı yazmassanız programı çalıştırdığınızda alacağınız hata :

QSqlDatabase: QMYSQL driver not loaded
QSqlDatabase: available drivers:

şeklinde oluyor. Yani QMYSQL sürücüsü yüklü değil ve mevcut driver da yok. Bu sebeple hiçbir qt uygulamanızda QAplication nesnesi oluşturmayı unutmayın ;) Nesne oluşturduktan sonra conect_data() fonksiyonunu if parantezi içinde çalıştırarak bağlantı sağlayamamız durumunda olası bir hatayı engelliyoruz.

QtSql modülü içerisinde sql deyimlerini çalıştırmak ve veri işlemek için **QSqlQuery** sınıfını kullanıyoruz. Bu işlemleri yapmak için QSqlQuery türünden bir nesne oluşturmak ve exec fonksiyonunu istediğimiz sql deyimi ile çalıştırmaktan başka bir işlem yapmaya da gerek yok. Tüm deyimler için (insert, update, select, delete) QSqlQuery nesnesi kullanabiliyoruz. Yukarıdaki örneğimizde ilk sql deyimi ile tablomuzda alan2 'ye "ondel" değeri giriliyor, ikinci sql deyimi ile "ondel" değeri "onder" ile değiştiriliyor, üçüncü deyimde tüm kayıtlar seçiliyor ve dördüncü deyimde alan2 de "onder" olan kayıt(lar) siliniyor. Verilerin ekrana yazdırılmasını bir while döngüsü ile (bu örnekte) yapıyoruz. select sorgusunun sonucu next fonksiyonuna doğru değeri döndürdüğü sürece yani kayıt olduğu sürece dönen bir döngü içinde value() fonksiyonunun değerini alarak ekrana yazdırıyoruz. qt ile veritabanı işlemleri bu kadar kolay ;)

Derleme ve Çalıştırma

Program derlenip çalıştırıldığında aşağıdaki gibi görünür :

```
onder@kute datatest $ ./datatest
23 onder
```

Tabii siz programı ilk çalıştırdığınızda muhtemelen alan1 'in değeri 1 olur ben 23 kez çalıştırdığım için

alan1 'in auto_increment özelliğinden dolayı değeri 23 ;)

Son Söz

Programlarınızı derlerken dikkat etmeniz gereken bir nokta var eğer QtSql modülünü kullanıyorsanız qmake komutundan sonra oluşan .pro dosyanıza

```
QT += sql
```

satırını eklemek zorundasınız. Aksi takdirde make komutunda QSqldatabase, QSqlQuery gibi sınıfların tanımlı olmadığına dair hata mesajları alırsınız.