Subject: Re: U++ SQL Begginer
Posted by captainc on Thu, 07 Aug 2008 13:23:01 GMT
View Forum Message <> Reply to Message

Ahh, I just learned this, but there is only 1 SQL object being used in the background to manage all the sessions.

From sqls.h:
```
struct AppSql : Sql {
 void   operator=(SqlSource& s) { Assign(s); }
 void   Detach()              { Sql::Detach(); }
 AppSql() : Sql(NULLSQL) {}
};
AppSql& AppCursor();
#define SQL AppCursor()
```

When you make an Sql object, that sql object you make will actually call the global SQL object
For example, from sql.cpp:
```
Sql::Sql(const char *stmt) {
 cn = SQL.GetSession().CreateConnection();
 SetStatement(stmt);
}
```

The sqlite3 example you are referencing shows an alternate way of working with the sql packages, calling on the SQL object itself. Instead, to keep it all similar, do something like this:
```
Sqlite3Session m_session;
bool good_conn = m_session.Open("my_database_file.db");
Sql sql(m_session); //define Sql object to act on Session object m_session.
```
Notice how I used Sqlite3Session just like I used PostgreSQLSession in the tutorial example. Also, I passed the session to the Sql object as well. Here we used the Sql object's constructor instead of assigning the sqlite3 object to the global SQL object.

```
Sqlite3Session m_session;
Sql sql(m_session);
```

- instead of -

```
Sqlite3Session sqlite3;
SQL = sqlite3;
Sql sql;
```

Other than that, it should otherwise be all the same.