Subject: Re: Firebird support
Posted by AJM on Wed, 02 May 2007 10:04:29 GMT
View Forum Message <> Reply to Message

1. There are numerous places where tables from different databases (or â€œschemasâ€• in PostgreSQL and Oracle) are accessed at the same time, sometimes within a JOIN but mostly through their separate classes. The MENU database is accessed within every script in order to verify the userâ€™s access to the selected task, and to extract the information necessary to build the menu bar and the navigation bar. The AUDIT database is accessed during every database update to record was has changed. It would be possible to move all the tables into a single database, but it would take a tremendous amount of work. All the while I have been building administrative applications, which is over 20 years now, it has been standard practice to give each subsystem its own database and its own code area. This can be handled with MySQL, PostgreSQL and Oracle, but if Firebird cannot support separate databases or schemas then it sounds like it is not suitable for use with Radicore.
2. Radicore allows you to construct whatever sql queries you like, and the performance of those queries is outside Radicoreâ€™s control.
3. It is not necessary to use *_free_result or *_free_statement all the time as all resources are automatically released when the script ends. Where the resource is potentially large, e.g. after a getData(), it is released as soon as it has been processed, but in other cases I do not bother. This does not cause a problem, so I have no plans to change my code.
4. It is standard practice to have no more than one autoincrement column in a table, and to use that within the primary key. I have constructed the MySQL, PostgreSQL and Oracle drivers accordingly. Anybody who uses more than one autoincrement column needs to rethink their database design.
5. All fatal errors are routed through the error handler which is contained with file error.inc, and if you look you will see that if a transaction is in progress it will automatically perform a rollback. I have no plans to change Radicore to use exceptions for the simple reason there would be no benefit. The existing system works well, and changing it would not make it work any better. The Radicore code base works with PHP 4 which does not have exceptions, so that rules it out completely.
6. Database locking is described in http://www.tonymarston.net/php-mysql/infrastructure-faq.html#faq48
7. Different databases may have different or variable transaction levels, and how these are implemented is the responsibility of the individual database driver.
8. ---
9. It is not up to the framework to validate that an SQL query that you have generated is either valid or efficient. That is entirely up the developer.

I have never read any Microsoft articles on multi-tier development as they are oriented around Microsoft products which I do not use. I first read about the three-tier architecture when I was working with an obscure language called Uniface. The team I was working with designed a framework that was so horrendous the client cancelled the contract. I built my own version in two weeks (by modifying my existing two-tier framework) which outperformed what had taken them three man-years. The problem was that their framework was based on theory while mine was based on practical experience. Ever since that moment I have stopped listening to these â€œexpertsâ€• with their wild theories and done things my own way. The result is Radicore.