# Firebird & MySQL - Parceiros ou Concorrentes?

Muito se tem falado sobre a rivalidade entre os SGBDs, particularmente entre os open-source. Afinal, se são gratuitos, o fator "custo" não é mais um componente da escolha.

Sobre o quê deve recair então a escolha? E afinal, quem é o melhor entre o Firebird e o MySQL? Eles são realmente concorrentes?

## Um pouco de história

Antes de entrarmos na discussão técnica, é importante conhecer um pouco da história destes dois SGBDs. Isto pode nos ajudar a entender as diferenças atuais nos estágios dos projetos.

O Firebird é oriundo do código do Interbase 6 open-source. Portanto, quando falarmos no Firebird, temos de lembrar da história de mais de 15 anos de desenvolvimento do Interbase. O Firebird é o sucessor do Interbase open-source, uma vez que a empresa fabricante do Interbase, a Borland, decidiu fechar o código nas versões seguintes. O Firebird conta com um código aperfeiçoado, livre dos bugs do IB 6 e com recursos adicionais na versão 1.0. Atualmente encontra-se em testes a versão 1.5 Alfa do Firebird, que trará importantes avanços técnicos. O Firebird é mantido por uma comunidade internacional de desenvolvedores, sem vínculo entre si, ou nenhuma empresa patrocinadora. Ele realmente depende do esforço voluntário da comunidade para continuar a crescer.

O MySQL nasceu há menos tempo, inicialmente como um produto proprietário, logo em seguida apenas com restrições na licença para uso comercial, mas que com o tempo adotou a GPL (General Public Licence). Neste intervalo de tempo, com os recursos captados, foi viabilizado o investimento no desenvolvimento do produto.

Ele em seguida tornou-se popular por ser distribuído juntamente com as compilações mais populares do Linux, como Red Hat, Mandrake, Suse e Slackware. A repercussão foi enorme, tornando a empresa original por trás do MySQL bastante sólida, que atualmente oferece o MySQL livremente junto com uma versão paga, mais poderosa que a versão free. Portanto há uma empresa com recursos financeiros suficientes para garantir o fôlego por trás do MySQL, além do apoio explícito de muitas outras. Considerando-se este fator, podemos até afirmar que o grau de evolução do MySQL deveria ser maior.

Neste artigo estaremos estabelecendo as conparações entre o Fierbird e a versão free do MySQL, para evitar colocar nesta discussão o Firebird contra produtos pagos, que renderia mais umas dúzias de páginas.

## Rumos e objetivos distintos

Comparar as duas soluções desconsiderando-se as necessidades do usuário, seria uma injustiça com qualquer um dos projetos. Ambos tem virtudes e limitações que podem ser interessantes para uns e desfavoráveis para outros. É quase como religião, política ou futebol: cada um tem suas crenças, e as defende com muita disposição.

O Firebird é conhecido por qualidades notáveis, com características de um SGBD estável, seguro e robusto. O MySQL se notabilizou pela grande velocidade e especialização em ambiente WEB. Portanto dependendo dos objetivos a serem alcançados você pode preferir um ao outro.

**Tamanho é documento?**

As diferenças entre os dois começam no tamanho da solução: a instalação do MySQL ocupa em torno de 12Mb, enquanto o Firebird não passa de 2,5 Mb.

O suporte à multiplas plataformas é ponto comum entre as duas soluções, levando o Firebird uma ligeira vantagem.

**Ferramentas**

O MySQL tem um número mais limitado de ferramentas de acesso e manutenção às bases de dados, ao passo que o Firebird, por ter o mesmo "berço" do Delphi e C++, conta com um número maior de ferramentas, propiciada pelo alto grau de afinidade com estas linguagens, bastante populares.

As opções mais conhecidas e confiávies de ambiente gráfico do MySQL são o MySQLAdm o MySQLGUI enquanto no Firebird temos o IBExpert, IBOConsole, Firebird Workbench e muitas outras.

**Interface com Linguagens**

Neste aspecto, fica muito evidenciado a tendência do MySQL no suporte à Web e do Firebird à aplicações desktop. O Firebird é contemplado nativamente em diversas linguagens, particularmente Borland, embora através do ODBC possa-se ligá-lo virtualmente a qualquer "coisa".

O MySQL tem uma "simpatia" maior pelas linguagens da comunidade open-source como Perl, Phyton e PHP, embora estas barreiras não existam mais com o Firebird.

**Armazenamento**

O modo como as informações são armazenadas difere bastante. Isto tem um impacto direto em questões como backup, integridade dos dados e segurança.

O Firebird armazena todo o conjunto de de dados, além dos procedimentos ligados ao banco de dados, gatilhos disparados automaticamente e visões em um único arquivo. O MySQL por padrão guarda as tabelas em diversos arquivos, usando-se um esquema de 3 arquivos para cada tabela e todos os aquivos do banco de dados ficam dentro de um diretório.

Digamos que você trabalhe numa empresa que tem 4 filiais e diariamente você precisa atualizar a tabela de produtos para as filiais. Com o MySQL basta copiar os arquivos da tabela de produtos, que seriam: PRODUTOS.FRM, PRODUTOS.MYD, PRODUTOS.MYI. Há quem diga (e eu concordo) que isto é uma coisa que jamais se deve fazer.

Com o Firebird, a única saída será você  criar um programa para fazer isso. Para se fazer uma cópia de tabela você é obrigado a usar as ferramentas do banco, backup e restore, para o SGBD garantir a integridade dos dados. Para quem foi “Clippeiro” ou desenvolvedor Paradox isto representa ter que repensar o “modo operantis” de suas aplicações. O impacto é um pouco menor para quem usa MS-Access, pois conceitualmente são parecidos.

Os puristas do mundo SGBDs chegam a afirmar que o MySQL ainda não se tornou um SGBD de verdade, pois não tem nas suas características nativas este e outros recursos importantes.

## Padrão ANSI SQL 92

Por ser totalmente compatível com o padrão ANSI SQL 92 o Firebird implementa uma série de funcionalidades nativas, que no caso do MySQL não estão disponívies, algumas podendo ser alcançadas com ferramentas de terceiros:

- Integridade Referencial e outras constraints.
- Subselects
- Unions
- Views
- Triggers
- Stored Procedures
- UDF (User Defined Functions)
- Suporte à transações

Há previsão de alguns destes itens serem implementados em versões futuras do MySQL, alguns inclusive em beta teste, e outras como já dito, com o auxílio de ferramentas, que permitem modificar a maneira como o MySQL armazena os dados, como o InnoDB, que assegura um modelo transacional no MySQL. Na configuração padrão o MySQL é apenas um gerenciador de arquivos com suporte à SQL.

Estas diferentes possibilidades de configuração do formato das bases de dados com o MySQL apesar de parecerem interessantes, podem representar dificuldades na hora de integrar bases de dados com padrões diferentes, ou transportar os dados de um servidor para outro. No Firebird, por haver um padrão único, independente dos recursos utilizados, o transporte entre diferentes servidores não é um problema.

Vamos dar uma olhada no custo-benefício de cada uma destas características:

## Integridade Referencial, bom para quem?

Manter os dados que se relacionam íntegros pode ser crucial para o seu negócio. Por exemplo, imagine uma empresa que guarda todos os pedidos de um cliente, e este cliente "some" do banco de dados. Isto deve ser evitado, mas pode ser feito de várias maneiras, não necessariamente pelo banco de dados, embora o custo disto seja maior em termos de tráfego e processamento pela aplicação.

No Firebird este é um conceito natural, no MySQL depende do sistema de arquivos utilizado. Para quebrar a integridade referencial do MySQL, basta copiar uma versão antiga de uma das tabelas sobre a atual, e está feito o estrago.

Mas digamos que você vá utilizar o banco de dados apenas para disponibilizar uma lista de preços ou um cadastro de pessoas simples. Para que integridade referencial? Não é necessário.

## Subselects

Um recurso interessante do Firebird, que demanda menos programação para extrair-se dados mais complexos.

## Unions

Montar consultas com Unions dá ao programador um poder maior de composição dos dados, coisa que deve ser contornada no MySQL com a criação de tabelas temporárias, em disco ou memória.

**Views**

Muito mais que maneiras de visualizar dados, as views são uma poderosa maneira de implementar segurança, performance e recursos de consultas avançadas, pois sobre uma view pode ser aplicado um Select de alta complexidade.

O MySQL fica devendo este recurso aos desenvolvedores, fragilizando a base de dados por torná-la totalmente exposta.

**Triggers**

Ao descobrir o quanto se pode poupar de código com o uso de Triggers, o seu uso torna-se altamente justificável. Através de gatilhos disparados automaticamente quando os dados são manipulados (inseridos, alterados ou apagados).

Por exemplo, você pode criar triggers que atualizem automaticamente o saldo de uma conta contábil, ou o saldo de estoque quando é feita uma movimentação.

No MySQL isto precisa ser feito nas aplicações, e imagine o "custo" disto se você tiver 30 diferentes módulos que fazem este movimento: serão trinta locais diferentes onde será preciso dar manutenção no código.

**Stored Procedures**

Ainda em fase de testes na última versão beta MySQL, é um recurso nativo do Firebird. Através do uso de Stored Procedures, pode-se criar processamentos complexos sobre os dados, com passagem e retorno de parâmetros. Assim como no caso das Triggers, com o uso de Stored Procedures, você transfere código da aplicação para o banco de dados, poupando tráfego e processamento pela aplicação no lado cliente.

**UDF (User Defined Functions)**

Imagine tudo aquilo que você gostaria que o Firebird fizesse, mas não faz. Sem problema. Basta você criar uma biblioteca de funções que poderão ser usadas pelo seu banco de dados para extender as funcionalidades do Firebird.

O MySQL depende totalmente da aplicação para agregar recursos, não sendo possível ampliar sua funcionalidade senão por meio dos back-ends disponíveis.

**Suporte à transações**

O MySQL pode dispensar o uso de transações para manipulação de dados. No Firebird isto é impossível. Mesmo uma simples consulta ocorre dentro do contexto de uma transação. É por isto que dependendo da atividade isto pode tornar o Firebird um pouco mais lento. Afinal, sempre haverá a abertura de uma transação e o encerramento, para qualquer coisa feita. Isto gera um GAP muitas vezes indesejável, mas o benefício disto é a segurança oferecida.

**Firebird, MySQL e a corrupção de dados**

Nenhum SGBD é totalmente à prova de corrupção, até porque dependem muito da plataforma utilizada, da confiabilidade do Hardware, da qualidade da energia elétrica e até da boa vontade dos administradores do banco de dados.

Parece que por oferecer uma gama de recursos menor, fazendo portanto menos controles, o MySQL tende a ser menos suscetível à corrupção. Mas isto é muito discutível, e há vários casos relatados de corrupção em ambos, todos oriundos de

plataformas deficitárias.

O fato é que nenhum dos dois tem detecção automática de corrupção de dados.

**Performance x Recursos x Segurança**

Esta é a grande chave para a sua escolha. Nenhum dos dois é perfeito nos três itens. Em certos momentos pesará mais um fator do que outro. Identificar isto é fundamental, além é claro de ter a noção do trabalho para implementar cada projeto, isto é, lembre-se que tudo aquilo que o SGBD não puder fazer por você você terá de criar se precisar.

O MySql é muito conhecido pela sua performance em aplicações Web. De fato, de um modo geral, fora do mundo Web, ninguém leva o MySQL muito a sério, preferindo outras soluções.

O Firebird oferece maiores recursos e segurança, sendo o preferido em aplicações Desktop e onde a segurança e integridade são cruciais.

Este é o cenário atual, mas ambos estão caminhando na direção das virtudes um do outro, buscando suprir soluções para estas questões. Por exemplo, existem muitas aplicações para construção de Web Sites em PHP que oferecem suporte ao MySQL, e dificilmente ao Firebird (PostNuke, PHPNuke, PHPBB, SlashCode, PHPMyAdmin). No entanto isto está aos poucos mudando, pois a necessidade de mais segurança em aplicações Web está levando o MySQL a um limite, então veremos logo uma mudança a favor do Firebird neste sentido.

**Mas afinal, porque o MySQL é mais popular que o Firebird?**

Este é um fato inegável. O MySQL tornou-se a grande "vedete" do confuso e imprevisível mundo open-source. Mas não é difícil descobrir o porquê disto:

- O MySQL veio praticamente agregado à todas as distribuições Linux importantes. Como o Linux teve seu "boom" justamente à 3 anos atrás, ele veio na "carona". Nesta época o antessor do Firebird ainda era um produto comercial.
- A ênfase dada à produção WEB no MySQL foi muito ampla, a maioria dos web sites que um visitante comum acessa, tem um logo com algo assim: "Powered by PHP & MySQL". Isto criou um interesse particular nestas ferramentas.
- O MySQL oferece uma migração mais amigável para quem vem da arquitetura desktop, pois ele não precisa trabalhar necessariamente com o modelo client-server, o que é um paradigma a ser quebrado para quem vem destes bancos de dados.
- O antecessor do Firebird sempre foi um produto secundário para o fabricante, que sempre concentrou-se em divulgar seus compiladores. O marketing foi portanto péssimo, à despeito de sua irrefutável qualidade.
- Ao nascer o Firebird não ficou vinculado à nenhuma empresa com recursos para investir em marketing, promoção ou ferramentas, sendo mantida por uma comunidade bastante ativa, porém extremamente voltada à parte técnica, e pouco direcionada à ações de marketing.

**Isto não torna o Firebird inviável?**

De maneira nenhuma! Em primeiro lugar, aos poucos este cenário está mudando, pois com a adesão de mais e mais desenvolvedores de grandes e pequenos projetos, o Firebird está ganhando mais visibilidade. Toda a comunidade de usuários

do Interbase está optando pelo Firebird, apostando no seu futuro. Isto dá ao Firebird um “lastro” muito interessante no que diz respeito aos seus “clientes”.

O Firebird está evoluindo muito rapidamente, após um início turbulento, muitos afirmando que era um projeto fadado ao declínio, em poucos meses surgiu a versão 1.0 e logo teremos a versão 1.5.

O Time de desenvolvedores Firebird, está ganhando a adesão de comunidades organizadas ao redor do mundo, como por exemplo, a CFLP (Comunidade Firebird de Língua Portuguesa) que estão fomentando a divulgação do Firebird, não só em caráter técnico, mas concedendo uma oportunidade de pessoas e empresas conhecerem melhor as vantagens de utilizá-lo em seus projetos, e tornar conhecidos profissionais e projetos "powered by Firebird”.

Esta mesma comunidade está fazendo um esforço para quebrar as barreiras da imprensa especializada, trilhando um longo caminho para mostrar-lhes a grande valia do Firebird, um verdadeiro processo de "doutrina" e persistência. Há negociações também no sentido de disponibilizar o Firebird junto à distribuições Linux.

## Conclusões

Há dezenas de motivos para adotar o MySQL. Há centenas de motivos para utilizar o Firebird. Dependendo do que você espera do banco de dados, a sua escolha recairá sobre um ou outro.

Quem gosta de um banco "seco", enxuto, fazendo tudo "no braço" talvez terá mais sucesso com o MySQL. Já quem gosta de ter mais poder em um SGBD relacional, preferirá o Firebird.

Uma coisa é certa: O Time de Desenvolvedores do Firebird quer torná-lo mais popular e o pessoal do MySQL reconhece que faltam recursos “da pesada” a serem desenvolvidos. No entanto se compararmos friamente o estágio evolutivo de cada produto, o Firebird “largou na frente” graças ao Interbase. Na questão de investimentos *versus* recursos, o Firebird também leva vantagem, pois tem um orçamento de “zero doláres” e mesmo assim segue evoluindo num ritmo constante. Esperava-se que o MySQL já estivesse mais avançado, porém parece que os investimentos concentram-se mais no marketing.

Por gostar do MySQL e ser apaixonado pelo Firebird, tenho usado o MySQL em situações bem específicas, e deixando para o Firebird a maior parte dos projetos. No entanto, não os vejo como concorrentes, pois servem bem às suas finalidades, tornando-se parceiros, com funcionalidades e capacidades diferentes, que se ajustam por afinidade a cada necessidade.

# Anexo I – FAQ para usuários que estão migrando do MySQL

- **Como Instalar o Firebird?**
- **Como conectar-se à um banco de dados Firebird?**
- **Há alguma ferramenta para administração em PHP como o PHPMyAdmin para o Firebird?**
- **Como criar campos auto-incrementáveis no Firebird?**
- **Há algo para limitar registros retornados em um Query de Select?**
- **Como listar todas as tabelas de um banco de dados?**
- **Como contar o número de registros recuperados?**
- **Há algo mais que eu deveria saber?**

## Como Instalar o Firebird?

Para fazer o Download do Firebird, vá ao seguinte endereço na Web:

http://firebird.sourceforge.net/index.php?op=files

Na plataforma Windows, basta rodar o executável SETUP. A instalação na plataforma Linux é extremamente simples com o uso de pacotes .RPM. Para mais detalhes de instalação na plataforma Linux, vá ao endereço:

http://firebird.sourceforge.net/index.php?op=doc&id=install

## Como conectar-se à um banco de dados Firebird?

No MySQL todos os aliases são registrados no servidor. O Firebird 1.0 não implementa este recurso, então é preciso apontar o servidor e o banco de dados com todo o caminho ao arquivo de banco de dados. Exemplos de strings de conexão:

**MySQL:**

*SERVER_HOST=localhost*

*DATABASE_NAME=testdb*

*USERNAME=milanb*

*PASSWORD=******

**Firebird:**

*DATABASE_PATH=localhost:/usr/local/db/testdb.gdb*

*USERNAME=milanb*

*PASSWORD=*****

Como você pode ver, você se conecta ao servidor e seleciona o banco de dados ao mesmo tempo. O Firebird 1.5 (em fase de testes) suporta o uso de alias no lado servidor. Ao contrário do MySQL que armazena cada tabela e indíces em arquivos separados, o Firebird armazena todo o banco de dados em um único arquivo, que normalmente tem a extensão .gdb.

## Há alguma ferramenta para administração em PHP como o PHPMyAdmin para o Firebird?

Sim, existe, é chamada de WebAdmin e você pode baixá-la em:

http://sourceforge.net/projects/ibwebadmin/

Baixe o pacote e descompacte-o em um local acessível pelo servidor Web, altere o arquivo configuration.inc.php no diretório inc.

*Notas de instalação para usuários Windows:*

Se você utilizou os diretórios padrão do Firebird para instalação, etas são as entradas que você deve alterar:

*define('BINPATH', 'c:/progra~1/firebird/bin/');*

*define('SECURITY_DB', 'c:/progra~1/firebird/isc4.gdb');*

*define('PATH_SEPARATOR', '\\');*

ATENÇÃO: esta ferramenta ainda está em versão Beta, mas pode ser útil no seu dia-a-dia. Entretanto você pode e deve dar uma olhada em outras aplicações gráficas (comerciais, freeware e open-source) que estão estáveis e facilitam o uso do Firebird.

**Como criar campos auto-incrementáveis no Firebird?**

Para fazer isto, o Firebird usa Generators (geradores) como no Oracle. Cada Generator tem um valor. Abaixo um exemplo de como utilizar Generators em comparação com o MySQL:

**MySQL:**

```
CREATE TABLE test
(
field1 integer not null auto_increment,
field2 char(10),
PRIMARY KEY (field1)
);
```

inserindo valores:

```
INSERT INTO test (field2) VALUES ('testme');
```

**Firebird:**

```
CREATE TABLE test
(
field1 integer not null,
field2 char(10),
PRIMARY KEY (field1)
);

CREATE GENERATOR gen_test_id;
```

inserindo valores:

```
INSERT INTO test (field1, field2) VALUES (gen_id(gen_test_id, 1), 'testme');
```

Isto pode parecer um pouco complicado, mas os Generators lhe dão muito mais poder que campos *autoinc* uma vez que você pode recuperar o valor de um Generator sem incrementá-lo:

*SELECT gen_id(gen_test_id, 0) FROM ...*

E você pode modificar o valor atual com:

*SET GENERATOR gen_test_id TO [algum_valor];*

Para saber mais sobre Generators consulte o Data Definition Guide do Interbase

(que se aplica ao Firebird) ou ainda o artigo disponível no site da CFLP.

**Há algo para limitar registros retornados em um Query de Select?**

Sim é um recurso chamado FIRST x SKIP y, e é usado desta maneira:

*SELECT FIRST x SKIP y FROM ... [resto da query]*

Isto irá retornar um total de x registros, pulando os primeiros y registros (iniciando-se do registro y+1)

**Como listar todas as tabelas de um banco de dados?**

No MySQL você utiliza o Show Tables. Você pode utilizar o mesmo comando do ISQL (utilitário de linha de comando que acompanha o Firebird), mas somente nele. Entretanto, isto pode ser feito fazendo uma consulta nas tabelas de sistema do Firebird:

*SELECT RDB$RELATION_NAME FROM RDB$RELATIONS;*

Esta query lhe mostrará as tabelas do sistema e de usuário (as que você criou). Para ver apenas as suas tabelas use:

*SELECT RDB$RELATION_NAME FROM RDB$RELATIONS WHERE RDB$SYSTEM_FLAG = 0;*

Nota: A consulta acima irá retornar além das tabelas de usuário, todas VIEWS criadas. Para exibir somente as tabelas use:

*SELECT RDB$RELATION_NAME FROM RDB$RELATIONS WHERE RDB$SYSTEM_FLAG = 0 AND RDB$VIEW_BLR IS NULL;*

**Como contar o número de registros recuperados?**

O Firebird não dispõe deste recurso. Você pode fazer isto trazendo todos os registros, ou utilizando um *SELECT COUNT (*)*... usando a mesma query.

**Há algo mais que eu deveria saber?**

Sim. O Firebird é um servidor maduro, e algumas de suas caracterísiticas podem lhe ser muito úteis, mas talvez você não as conheça:

Integridade Referencial

SubSelects

Unions

Triggers

Stored Procedures

UDF (User defined Functions)

Procure descobrir o poder que estes recursos lhe dão, consultando a documentação disponível, e torne-se mais um apaixonado pelo Firebird.

## Anexo II – O que andam dizendo por aí...

## Esta é a tradução de um artigo publicado na revista PC Week comparando o MySQL com outros bancos de dados relacionais:

### O MySQL oferece consulta de banco de dados mais rápida, mas pode deixar alguns usuários desejosos

**14 de fevereiro de 2000 12:00**

Qualquer avaliação de bancos de dados de código-aberto (open source) estaria incompleto sem uma discussão do MySQL, um banco de dados leve e magro que é um favorito de muitos Webmasters pela sua velocidade e tamanho pequeno.

Em particular, o desempenho rápido de leitura do MySQL (especialmente quando usado com um número pequeno de usuários), host name, subnet - ou camada de segurança de usuário baseada em nomes, e integração estreita com Perl e PHP torna uma combinação perfeita para uso da Web. Por exemplo, Slashdot.org é executado em um banco de dados de MySQL.

Entretanto, PC Week Labs acautela que o MySQL não deve ser comparado com bancos de dados de alto-desempenho, como o Firebird ou PostgreSQL (veja Análise da Tech). Este é fundamentalmente um produto diferente com objetivos de projeto diferente.

O MySQL é um banco de dados construído para retornar dados estáticos simples como informação de conta de usuário em Webpages tão rápido quanto disco e velocidades de CPU permitirão. Não é projetado para fazer—nem deveria ser usado para—transações financeiras, administração de inventário ou outras tarefas de negócio críticas. O ramo que o MySQL, junto com bancos de dados de Método de Acesso Seqüentes Indexados semelhantes, inclusive Microsoft Corp.'s Visual FoxPro e Corel Corp.'s Paradox, fazem é deixar a característica de banco de dados SQL de um alto-desempenho definindo — transação e histórico de transação — para ganhar simplicidade e velocidade.

Suporte de transação assegura que operações de banco de dados são atômicas, consistentes, independentes e duráveis (mais conhecido como ACID), garantindo assim que as atualizações em grupo de um banco de dados aconteçam corretamente e que o banco de dados possa sobreviver a deficiências, como por exemplo de disco ou de força-elétrica, corretamente. Nenhum negócio deveria pôr dados críticos internos ou dados de cliente em qualquer banco de dados que não dispõe operações ACID.

Prós do MySQL:

*- Operação muito rápida para aplicações de leitura-pesada ou de baixa carga de usuários*

*- Sistema de segurança flexível*

*- Forte Connectividade para sistemas de publicação de Web dinâmica*

Contras do MySQL:

*- Nenhum suporte à transação*

*- Sistema de travamento de tabelas se comporta pobremente quando há muitos*

*usuários competindo*
*- Incompatibilidade com o SQL-92*

O suporte à multi-usuário no MySQL também é bastante pobre — pode emitir apenas bloqueio a nível de tabela. Qualquer atualização no banco de dados trava a tabela inteira, fazendo com que todos as leituras aguardem até que a gravação seja concluída. Este pode ser um matador de desempenho, e o desenvolvedor de MySQL precisará escrever SQL muito cuidadosamente para evitar esta situação. (Em um caso especial, o MySQL pode permitir INSERTS simultâneos junto com leituras, contanto que os INSERTS só aconteçam no fim do arquivo — quer dizer, apenas que não haja nenhuma abertura vazia de registros apagados.)

Usando-se bloqueio em tabelas quando não há concorrência de bloqueio é um real ganho de velocidade — quanto mais bloqueios há, menos processamento de bloqueio é necessário. Em situações como na Web, onde há muita pequena concorrência de bloqueio, esta aproximação realmente vale. Porém, geralmente nós acreditamos que desempenho absoluto importa menos que desempenho previsível, que bloqueio mais granular – em nível de palavra, página ou à nivel de linha - provê.

O MySQL precisaria de um idioma de SQL característico, como subselects e visões, como também cursores do lado-servidor e procedimentos armazenados. Como resultado, nós não pudemos executar o mesmo código como ponto de referência que nós usamos com o Firebird e PostgreSQL com o MySQL.

Nós executamos há pouco uma transação on-line que processa teste de leitura-simples com um único usuário. O MySQL realmente foi fantástico, com um clocking de 649 transações por segundo, um resultado que é mais rápido que o processamento do Firebird aproximadamente quatro vezes e mais de 10 vezes mais rápido que o processamento do PostgreSQL à mesma carga de usuário e carga de trabalho. Nós planejamos refazer o código de ponto de referência para evitar as áreas de incompatibilidade de SQL do MySQL incluindo acesso multi-usuário estabelecendo um ponto de referência do MySQL depois, este ano.

Tradução e adaptação:

**Antonio Porfírio - (Tony)**
antonioporfirio@ipdal.com.br
Colaborador da Comunidade Firebird de Lingua Portuguesa

**Artigo Original**

**Paulo Vaz**
(Colaborador da CFLP)

*paulo@multi-informatica.com.br*

**Comunidade Firebird de Língua Portuguesa**

**Visite a Comunidade em:**

**http://www.comunidade-firebird.org**

A Comunidade Firebird de Língua Portuguesa foi autorizada pelo Autor do Original para divulgar este trabalho