

# Otimização de consultas no Firebird



## LEONARDO B. L. C. PIRES

(lbpires@gmail.com)

é graduando do Centro de Ensino Superior de Juiz de Fora - CES/JF no curso de Bacharelado em Sistemas de Informação, é um dos membros fundadores do JUGJF (Java User Group de Juiz de Fora) e atualmente vem trabalhando no desenvolvimento de soluções para dispositivos móveis e automação comercial na Hand Mobile Solutions ([www.hand.inf.br](http://www.hand.inf.br)).



## LUCAS REZENDE RICARDO

(lrezender@yahoo.com.br)

é graduando do Curso de Bacharelado de Sistemas de Informação no Centro de Ensino Superior de Juiz de Fora, atua como programador e membro da equipe de administração de dados da Zeus Rio Solutions ([www.zeusrio.com.br](http://www.zeusrio.com.br)), é um dos membros fundadores do JUGJF (Java User Group de Juiz de Fora), trabalhando com as ferramentas de desenvolvimento da Sybase, PowerBuilder e tecnologia Microsoft, incluindo banco de dados SQL Server.



## MARCO ANTÔNIO P. ARAÚJO

(maraujo@cesjf.br)

é Doutorando e Mestre em Engenharia de Sistemas e Computação pela COPPE/UFRJ, Especialista em Métodos Estatísticos Computacionais e Bacharel em Matemática com Habilitação em Informática pela UFJF, Professor do Curso de Bacharelado em Sistemas de Informação do Centro de Ensino Superior de Juiz de Fora, Analista de Sistemas da Prefeitura de Juiz de Fora.

Independente do tipo de aplicação desenvolvida, Web ou desktop, garantir o seu desempenho é fundamental. Para isso, é preciso analisar e entender os fatores que contribuem para degradação do desempenho de um sistema. Dentre esses fatores, um dos principais refere-se ao desempenho das operações executadas no banco de dados, principalmente das consultas, utilizadas com grande frequência.

Este artigo busca esclarecer algumas dessas questões relativas ao desempenho de consultas em bancos de dados e apresentar as ferramentas disponíveis no Firebird que auxiliam na tarefa de otimização, possibilitando a criação de consultas mais refinadas, objetivando o ganho de desempenho.

## Plano de execução

Quando uma operação é submetida ao banco de dados, seja ela um *Select*, *Insert* ou outra qualquer, o Firebird, assim como outros SGBDs, possui um mecanismo interno chamado otimizador de consultas, que tem a tarefa de analisar a operação a ser executada e avaliar a utilização de índices, *Unions*, *Joins* e outros, formando então um plano de execução, chamado de *Plan* no Firebird.

Como resultado do plano de execução tem-se o custo da operação, que é uma forma de avaliar se a consulta obteve um ganho de desempenho ou não. Internamente o otimizador executa uma seqüência de passos para escolher qual tipo de leitura será utilizado, se serão feitos diretamente sobre os registros ou através de estruturas de índices, bem como informar o tempo utilizado e o total de registros percorridos.

Cada consulta executada possui um plano específico que pode ser definido automaticamente pelo otimizador do Firebird ou manualmente pelo desenvolvedor da aplicação. Se esse otimizador avaliar que o custo da operação foi ruim, ele pode submeter a consulta a outra otimização ou então não usar otimização, utilizando um *Natural Plan*.

## Configurando um banco de dados para estudo

Para os estudos de otimização será utilizado o banco *Employee.fdb* que acompanha o pacote de instalação do Firebird

e encontra-se na pasta *examples* no diretório de instalação do banco de dados.

Como ferramenta para administração será utilizado o IBExpert que é uma ferramenta comercial para administração de bancos de dados da família InterBase e Firebird, mas possui uma versão funcional *free*, obtida no endereço [www.ibexpert.com](http://www.ibexpert.com).

Uma vez que o banco de dados já está criado, será necessário apenas registrá-lo no IBExpert para que seja possível executar as tarefas de administração. Os passos a seguir descrevem como registrar o banco no IBExpert.

Ao acessar o menu *Database>Register Database* um assistente como o da **Figura 1** será aberto. É nele que serão realizadas as configurações de registro do banco.

Em *Server* é preciso informar onde está instalado o servidor de banco de dados e possui duas opções: *Local*, no caso do servidor de banco de dados estar instalado na própria máquina que está trabalhando, e *Remote* para o caso de conectar a uma máquina servidora remota.

Quando escolher *Remote*, é preciso informar outros dois parâmetros que são *Server Name*, ou seja, o endereço da máquina servidora e *Protocol*, que é o protocolo para estabelecer a conexão com a máquina servidora. No caso deste exemplo, o servidor de banco de dados está situado na máquina local.

Quanto aos outros campos, *Database* e *Client Library File*, o primeiro refere-se ao caminho do arquivo de banco de dados que será registrado e o segundo representa a biblioteca utilizada para conexão com o banco.

O usuário administrador do banco de dados é o padrão, já criado quando é feita a instalação do Firebird, ou seja, o SYSDBA e a sua senha é “masterkey”. Depois de preenchidos os parâmetros de registro é preciso clicar em *Register* para registrar o banco de dados.

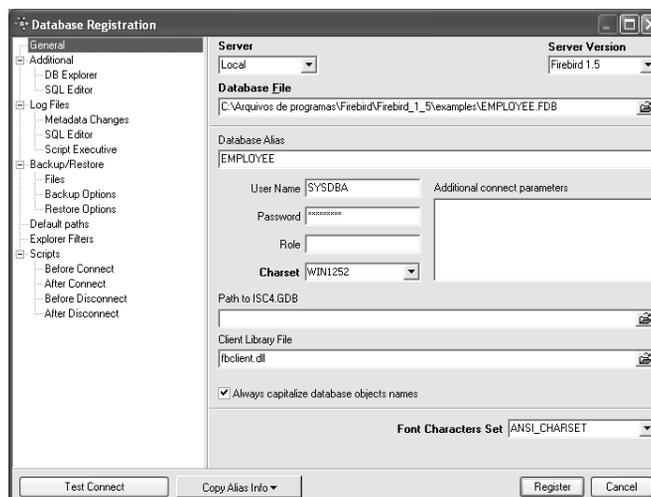
## Entendendo o Plano de Execução

Para um primeiro contato com o plano de execução do Firebird será executada uma consulta simples buscando listar todos os departamentos da tabela *Department* utilizando a ferramenta de execução de consultas do IBExpert, o *SQL Editor*.

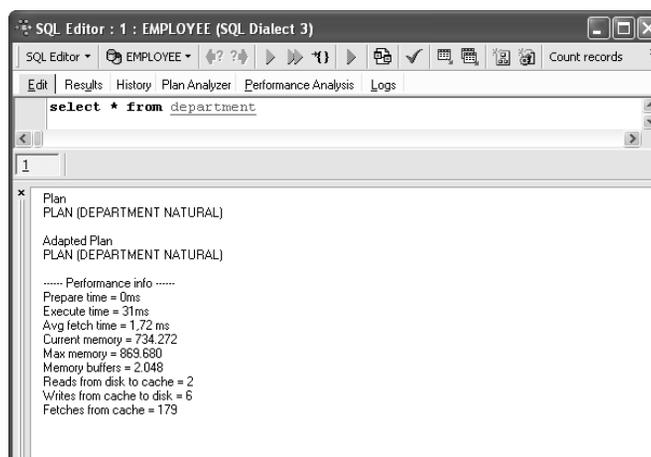
O acesso ao *SQL Editor* é feito através do menu *Tools>SQL Editor*. Com editor aberto, basta digitar o comando SQL: “select \* from department” e clicar no botão *Execute*, indicado por uma seta verde ou apertar a tecla de atalho F9 para executar a consulta (**Figura 2**).

A janela do *SQL Editor* é composta por diversas abas, sendo que, para os estudos de otimização, as mais utilizadas serão *Edit*, onde serão digitados os comandos SQL, *Plan Analyzer*, que informa em detalhes o plano de execução criado e executado para a consulta e *Performance Analysis*, responsável por exibir graficamente os detalhes do desempenho da consulta executada.

Na aba *Plan Analyzer* é possível visualizar com detalhes o *Plan* gerado para a consulta e os tipos de acesso que foram utilizados para fazer a leitura das tabelas envolvidas na consulta que, nesse caso, foi *Natural*, ou seja, uma leitura seqüencial sem a utilização de índices (**Figura 3**).



**Figura 1.** Assistente de registro de bancos de dados



**Figura 2.** Janela do SQL Editor

### ClubeDelphi PLUS

Acesse agora o mesmo o portal do assinante ClubeDelphi e assista a uma vídeo-aula de Luciano Pimenta que mostra como instalar e trabalhar com o IBExpert.

[www.devmedia.com.br/articles/viewcomp.asp?comp=3082](http://www.devmedia.com.br/articles/viewcomp.asp?comp=3082)

Na aba *Performance Analysis* é possível visualizar graficamente o desempenho da consulta. Nessa consulta o gráfico está representando uma leitura não indexada, ou seja, nenhum índice foi utilizado para leitura da tabela e, no final da barra, o total de registros percorridos é apresentado que, nesse caso, são 21 (**Figura 4**).

Na parte inferior da janela do *SQL Editor* encontra-se a aba *Messages* onde são apresentadas as estatísticas da consulta que foi executada e que são detalhadas a seguir:

- *Prepare time*: Tempo requerido para preparar a consulta;
- *Execute time*: Tempo de execução da consulta;
- *Avg fetch time*: Tempo médio de busca;
- *Current memory*: Memória utilizada;
- *Max memory*: Quantidade máxima de memória disponível por operação;
- *Memory buffers*: Tamanho do *buffer* de memória para armazenamento;
- *Reads from disk to cache*: Leituras feitas no disco e colocadas na memória;
- *Writes from cache to disk*: Escritas executadas em cache;
- *Fetches from cache*: Buscas no cache de disco.

Os dados apresentados em *Performance Info* são indicadores de desempenho, com destaque para o *Execute Time* que é o tempo de execução da consulta. Quando o *Execute Time* é elevado pode significar que o desempenho da consulta está comprometido.

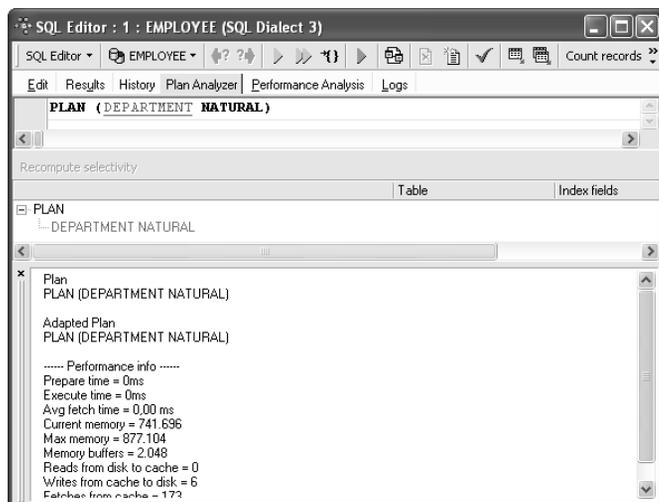


Figura 3. Aba Plan Analyzer

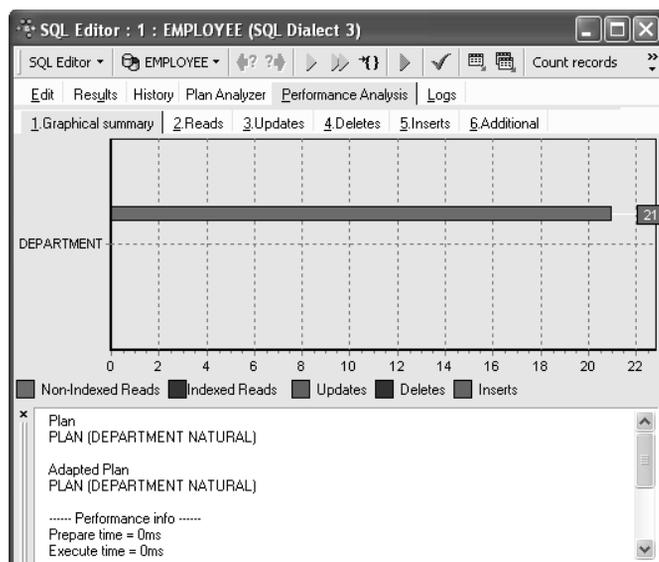


Figura 4. Aba Performance Analysis

Para todo *Plan* gerado pelo otimizador do Firebird a sintaxe básica será a palavra *Plan* acompanhada de expressões que identificará o tipo de acesso realizado ao ler as tabelas, que tabelas foram acessadas, se houve *Join* entre tabelas e a ordem de manipulação das mesmas.

Pegando como exemplo o plano gerado para listar todos os departamentos da tabela, a consulta e o plano gerado podem ser vistos na **Figura 2**. Nesse plano é possível observar duas expressões, além do comando *Plan*. A primeira é a tabela que está sendo lida, *Department*, e a segunda é o tipo de leitura na tabela, no caso *Natural*.

Quando temos a expressão *Natural* significa que a leitura foi feita de forma seqüencial, sem utilização de índices, ou seja, sem otimização. Em alguns casos não otimizar poderá ser uma melhor opção, uma vez que o custo para acessar o índice poderá ser maior do que uma leitura seqüencial, ou até mesmo devido à necessidade da consulta de acessar todos os registros mesmo que existam índices.

## Utilizando índices

Internamente, o banco de dados é armazenado em blocos de disco e, a cada consulta, as páginas que armazenam dados da tabela são varridas. Nessas páginas estão todas as linhas que o Firebird lerá para retornar o resultado. Essa varredura pelas páginas pode ser feita de maneira seqüencial, onde todas as linhas são lidas ou com a utilização de índices.

Os índices são usados somente quando são bastante seletivos, ou seja, o resultado trazido deve possuir no máximo 5% do total de linhas armazenadas. Toda essa análise é feita internamente pelo SGBD para que obtenha um ganho no custo estimado pela execução do comando.

Quando uma consulta é submetida, o otimizador do Firebird sempre verifica se existem índices que poderão ser utilizados com o intuito de reduzir o tempo de execução da consulta e, por sua vez, o seu custo. Basicamente o Firebird busca fazer uso de índices para executar três tipos de tarefas diferentes, são elas: comparações de igualdade, *Joins* entre tabelas e cláusulas de *Order By* e *Group By*.

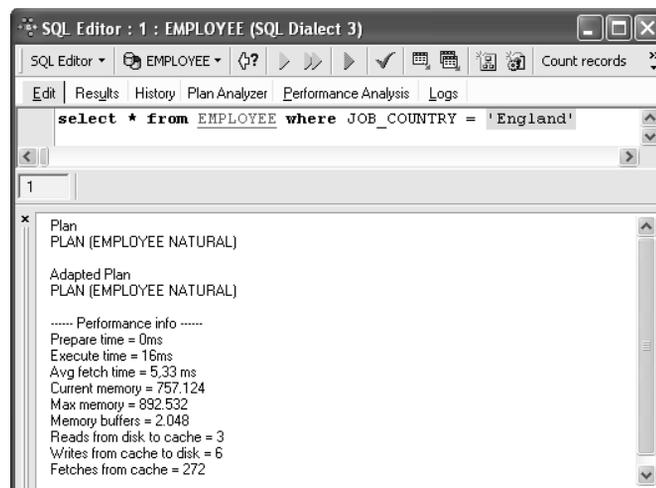


Figura 5. Consulta com cláusula where de comparação

Nas consultas que utilizam cláusulas de “maior que”, “menor que” ou “igual a”, ou seja, comparações de igualdade, os índices serão utilizados para remover registros que não cumprem a condição estipulada, evitando assim a leitura desnecessária desses.

Para exemplificar, foi elaborada uma consulta que utiliza da cláusula de comparação de igualdade *where* que lista todos os empregados da tabela *Employee* que trabalham na Inglaterra (*England*), com o seguinte código:

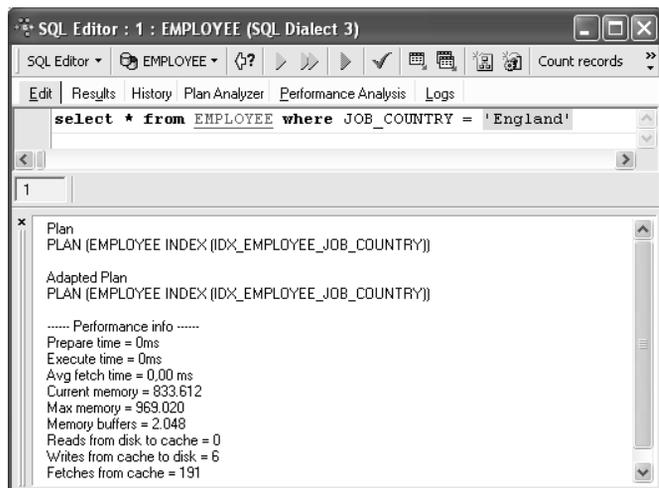
```
select * from EMPLOYEE
where JOB_COUNTRY = 'England'
```

Veja o resultado na **Figura 5**.

O plano gerado para essa consulta foi uma leitura seqüencial na tabela *Employee*, comparando quais registros cumprem a cláusula de filtro *where*. Como visto anteriormente, quando o tipo de acesso é *Natural* em uma tabela, significa que todos os registros foram lidos e comparados, o que certamente demanda uma quantidade de tempo maior do que acessar uma estrutura de índices para buscar os registros que satisfazem à condição.

Agora será criado um índice para a coluna utilizada na cláusula *where* que é a *Job\_Country*, com o seguinte código:

```
Create Index IDX_EMPLOYEE_JOB_COUNTRY on Employee (JOB_COUNTRY)
```



**Figura 6.** Consulta com o novo plano elaborado pelo otimizador

Para efetuar a criação do índice, é necessário executar o código anterior no *SQL Editor* e em seguida, executar um *Commit* na transação pressionando as teclas CTRL+ALT+C. Após criar o índice e executar novamente a consulta, é possível observar que o plano de execução mudou (**Figura 6**).

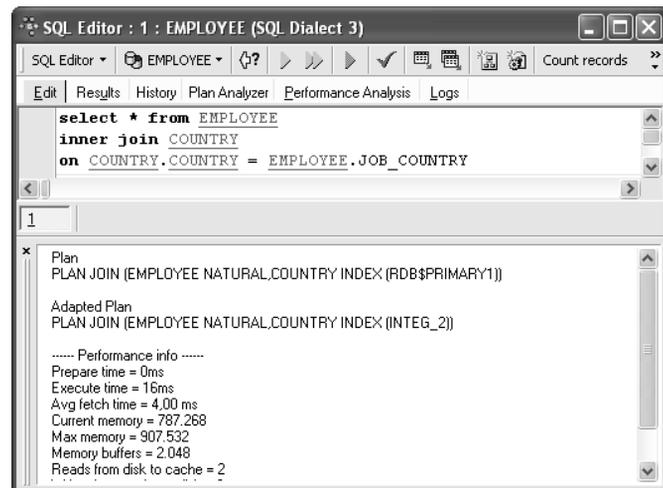
O tipo de acesso passou para *Index* e o tempo de execução diminuiu. A diminuição do tempo de execução é ainda afetada por diversos fatores, como a capacidade de processamento do servidor, o número de tabelas envolvidas, a quantidade de registros em cada tabela e o tamanho do registro.

Outra ocasião que faz com que o otimizador do Firebird opte pela utilização de índices é quando há a necessidade de realizar um *Join* entre tabelas, nesse caso os índices poderão ser utilizados pelo otimizador para realizar a operação com mais eficiência.

Quando existe pelo menos um índice em uma das colunas utilizadas na cláusula de *Join*, a coluna que não possui índices será lida primeira e os valores a serem encontrados serão procurados na estrutura de índices da outra coluna.

Para compreender melhor a questão da utilização de índices nos *Joins*, o índice para a coluna *Job\_Country* que foi criado no exemplo anterior será removido com o seguinte código:

```
DROP INDEX IDX_EMPLOYEE_JOB_COUNTRY
```



**Figura 7.** Consulta com Inner Join



Agora, execute a seguinte consulta:

```
select * from EMPLOYEE
inner join COUNTRY
on COUNTRY.COUNTRY = EMPLOYEE.JOB_COUNTRY
```

É possível observar que ao realizar o *Join* entre as tabelas *Employee* e *Country*, sabendo que a coluna *Country* da tabela *Country* é indexada, pois é a chave primária da tabela, e a coluna *Job\_Country* da tabela *Employee* não mais.

O otimizador optou por fazer uma leitura seqüencial na tabela *Employee* e procurar os valores na coluna indexada da tabela *Country* (Figura 7).

## Ordenação e agrupamento

Ao utilizar as cláusulas *Order By* ou *Group By* em colunas que possuem índices, o índice será utilizado para efetuar a ordenação ou agrupamento uma vez que o índice já está ordenado, não sendo preciso então realizar uma operação de *Sort*.

No exemplo da Figura 8 é possível observar uma consulta com uma cláusula *Order By* em uma coluna que não está indexada, sendo assim, será necessário executar uma operação de *Sort*, que aumentará o tempo de execução da consulta.

Execute o seguinte código para a criação de um índice:

```
Create Index IDX_EMPLOYEE_FIRST_NAME
on EMPLOYEE (FIRST_NAME)
```

Executando novamente a consulta anterior, dessa vez com a coluna indexada, é possível observar que o otimizador fez uso do índice para realizar a ordenação, uma vez que os índices mantêm a ordenação, melhorando o tempo de execução da consulta, poupando o otimizador de realizar ordenações que consumiriam um tempo maior (Figura 9).

Uma vez que o otimizador do Firebird tende a utilizar índices, é importante que esses estejam bem balanceados e que a sua seletividade esteja boa. Por isso, é importante que sejam realizadas manutenções nas estruturas de índices após alterações em grandes volumes de dados, principalmente quando essas alterações são do tipo *Insert* e *Delete*.

O balanceamento dos índices é feito através da sua reconstrução utilizando seguintes comandos:

```
Alter Index nomedoindice Inactive
Alter Index nomedoindice Active
```

O primeiro desativa o índice e o segundo ativa. Ao executar esse procedimento, de desativar e ativar o índice, ele será reconstruído e, por sua vez, rebalanceado.

Quanto à seletividade dos índices, ela é importante para auxiliar o otimizador na escolha do melhor caminho durante a execução de uma consulta, uma vez que os dados sobre a seletividade dos índices fazem parte dos dados estatísticos que o otimizador utiliza para formular os planos de execução.

Para atualizar a seletividade dos índices é necessário executar o seguinte comando:

```
Set Statistics Index nomedoindice
```

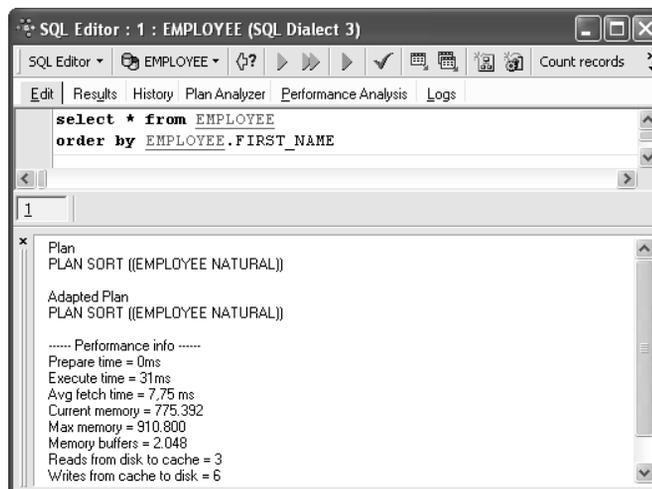


Figura 8. Cláusula de Order By em coluna não indexada

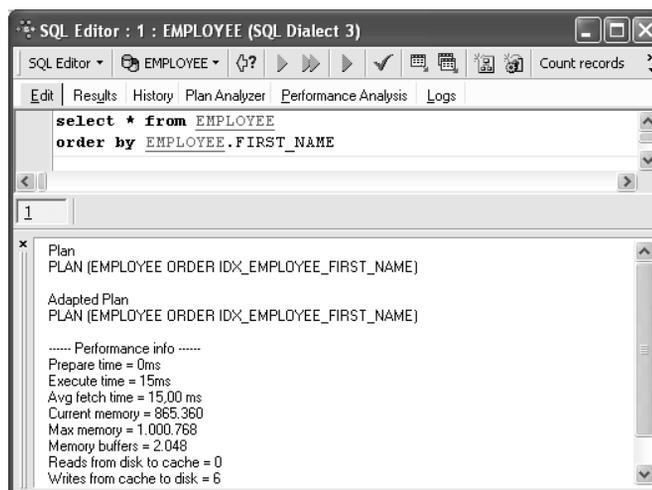


Figura 9. Consulta com cláusula de Order By na coluna indexada

Analisando os fatores apresentados anteriormente, é possível elaborar consultas que utilizam melhor os recursos que o otimizador do Firebird é capaz de fornecer.

## Conclusão

Apesar de que nos dias atuais, espaço para armazenamento ser quase que ilimitado, os recursos de rede e de processamento não são. Portanto, construir aplicações otimizadas poderá contribuir muito para um bom funcionamento e utilização do sistema desenvolvido, principalmente em aplicações cliente/servidor, em que requisições de dados ao servidor são feitas constantemente.

O importante ao realizar otimizações é saber que não existe uma só maneira que trará os melhores resultados. Técnicas de otimização possuem prós e contras que devem ser comparadas, principalmente a respeito das condições do ambiente onde o sistema está inserido, como rede, plataforma cliente/servidor e capacidade de armazenamento e processamento.

Assim, para todo e qualquer recurso, é importante avaliar exaustivamente as possibilidades para que não ocorram surpresas desagradáveis ao serem colocados sistemas em produção. ■