Definindo a arquitetura do Firebird

Este tutorial tem o objetivo de explicar as diferentes arquiteturas do Firebird, para servir como um guia de escolha.

O Firebird vêm em duas versões, chamadas de arquiteturas: *Classic Server* e *Superserver*. Desde o Firebird 2.5, o Classic Server pode operar em dois modos: "tradicional" *Classic* e *SuperClassic*, totalizando 3 modelos. Qual deles você deve escolher? As diferenças mais importantes estão listadas abaixo. Na grande maioria dos casos, todos os três modelos funcionam igualmente bem e oferecem (quase) as mesmas possibilidades.

1) Super Server

O *Super Server* é modelo padrão de arquitetura, e nele existe apenas um *cache* de páginas (dados em memória RAM) que é compartilhado por todas as conexões, além de um único processo atender a todos os usuários.

Por ser compartilhado, este *cache* é muito eficiente. Quando vários usuários acessam as mesmas áreas do banco de dados ou quando algumas tabelas são muito mais acessadas que outras, todos os usuários se beneficiam de um *cache* grande e bem preenchido.

Por exemplo, quando o usuário **A** executa:

```
SELECT NOME FROM CLIENTES WHERE CODIGO = 1;
```

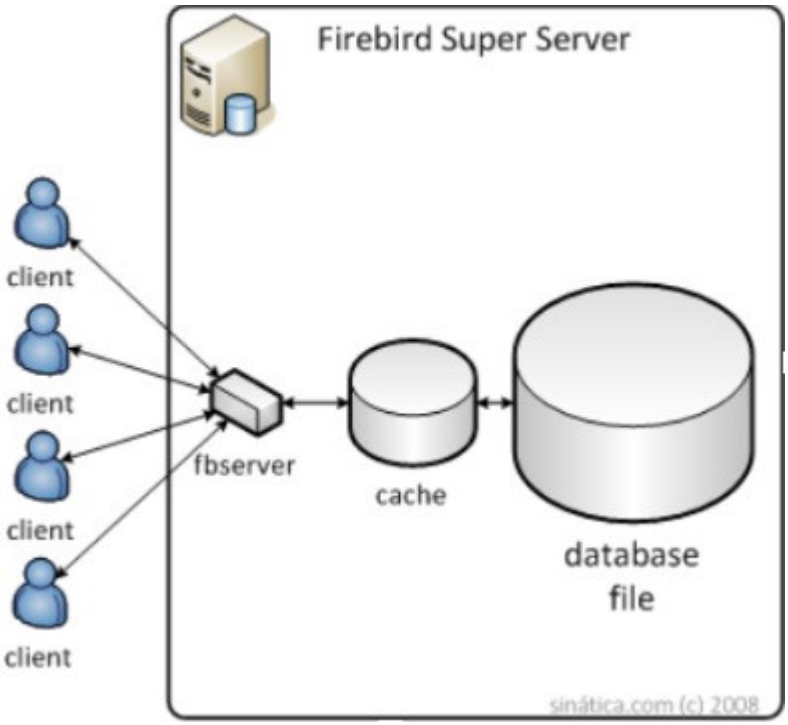
Algumas páginas relacionadas a tabela **CLIENTES** e ao índice da chave primária **CODIGO** são carregados para o *cache*.

Quando o usuário **B** executa:

```
SELECT NOME, ENDERECO, TELEFONE FROM CLIENTES WHERE CODIGO = 2;
```

A solicitação é atendida mais rapidamente, porque as páginas que este comando precisa consultar já estão no *cache*.

Veja no diagrama (considere cada “client” como um usuário do sistema):



Note também que existe apenas um único processo (*fbserver*) onde todos os usuários se conectam, e o importante aqui é que se um processo quebrar todas as conexões serão perdidas.

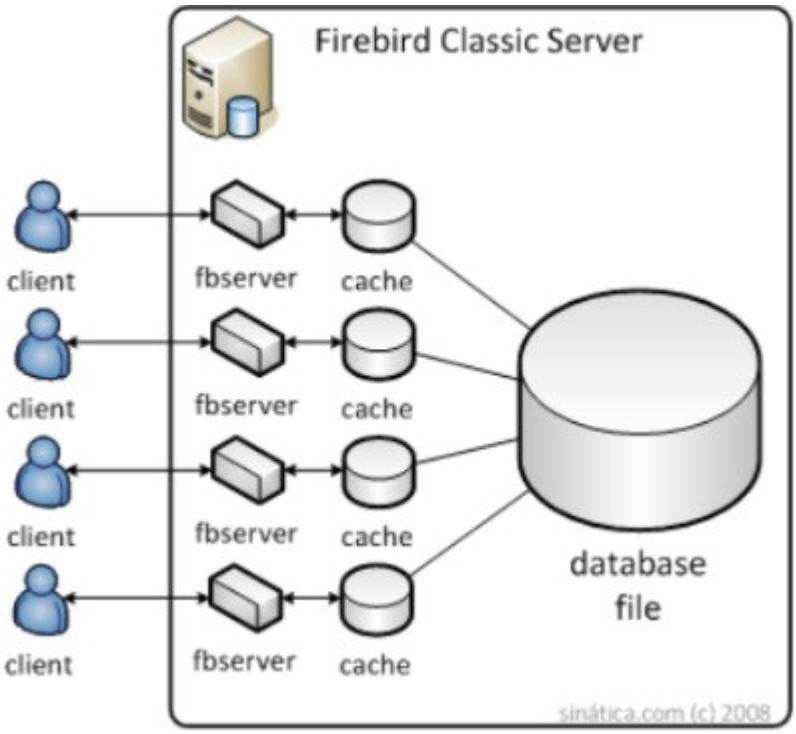
2) Classic Server

No *Classic Server*, cada usuário tem um *cache* próprio e está conectado a um processo dedicado.

O *cache* próprio é muito menos eficiente. Se dois usuário acessam a mesma área do banco de dados, esta área será copiada no *cache* de cada um deles. Usando o exemplo do item anterior, quando o usuário **B** executasse seu comando, ele não teria o benefício de um *cache* já preenchido e o Firebird precisaria acessar o disco novamente para responder.

Além do mais, a sincronização entre os *caches* é feita através do disco. Isto aumenta consideravelmente o custo de leitura/escrita em ambientes de alta concorrência.

Veja no diagrama:



Um grande benefício deste modelo é a resiliência oferecida pelos múltiplos processos. Se um deles tiver problemas, apenas o usuário conectado a ele será desconectado. Todo o restante do banco de dados continua funcionando normalmente.

O outro grande benefício é a escalabilidade. Acredito que esta característica seja a responsável por boa parte das instalações do *Classic Server*. Mesmo em casos onde o cache dedicado produz resultados inferiores ao cache compartilhado do *Super Server*, a escalabilidade compensa. Basta adicionar mais hardware e pronto, seu servidor fica mais rápido.

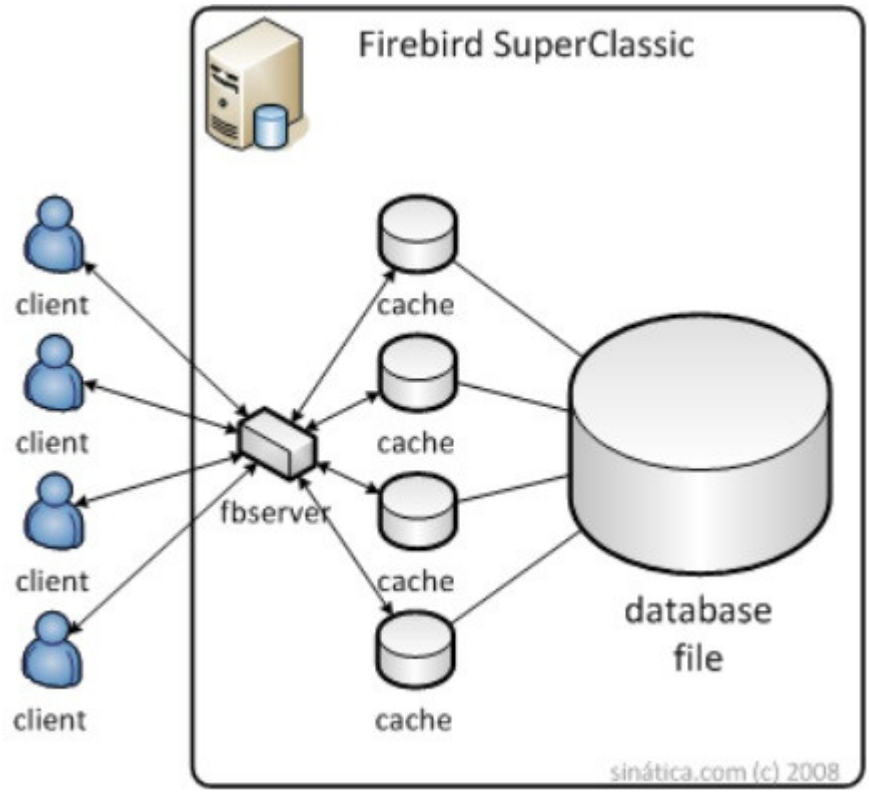
Mas esta escalabilidade não vem de graça. Imagine que você tem 200 usuários simultâneos. São 201 processos, um para cada usuário e mais um para ficar ouvindo novas conexões. Seu sistema operacional deve gerenciar todos estes processos e mantê-los em sincronia. Eles consomem muitos recursos de kernel e isto significa que ele pode ser relativamente lento. Imagine que um processo utilize +-60MB de memória, será necessário 12GB de memória disponível apenas para o Firebird.

Veja neste exemplo: Firebird 2.5 *Classic* com 7 usuários conectados. São 8 processos, 18 *threads* , 1050 identificadores.

Nome da Imagem	Identifi...	Threads
fb_inet_server.exe	133	2
fb_inet_server.exe	133	2
fb_inet_server.exe	134	2
fb_inet_server.exe	116	4
fb_inet_server.exe	135	2
fb_inet_server.exe	133	2
fb_inet_server.exe	133	2
fb_inet_server.exe	133	2

3) Super Classic

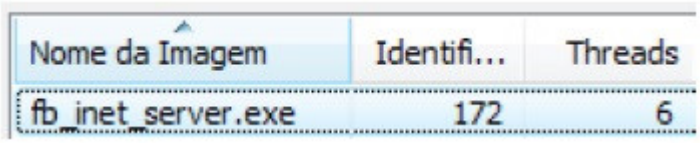
Super Classic é uma evolução e resolve o maior problema do *Classic*: todos aqueles processos pode deixar o Firebird lento e torna a manutenção mais difícil. Dessa forma o *Super Classic* opera em um único processo, assim como no *Super Server*, porém com *cache* próprio para cada usuário.



Olhando desta forma e considerando o nome, pode parecer uma arquitetura híbrida entre o *Classic Server* e *Super Server*, mas não é. O que fizeram foi colocar todos esses processos dentro de *threads*. Agora cada usuário tem uma *thread* dedicada dentro de um único processo.

Criar centenas de *threads* é muito mais barato do que criar centenas de processos e não há perda de escalabilidade. A sincronização de *cache* é feita diretamente na memória, o que reduz o custo de leitura/escrita. Outros controles que costumavam ser entre processos agora são entre *threads,* e muito mais rápidos.

Considerando o mesmo exemplo do item anterior, com 7 usuários conectados, nessa arquitetura são 1 processo, 6 *threads*, e 172 identificadores.



Conclusão

Esta compilação de casos mais comuns é uma sugestão e serve como guia, um ponto de partida para sua escolha. Sua implantação pode ter detalhes não contabilizados aqui.

Super Server

- Se você tiver 1 banco de dados principal e, opcionalmente, 2 a 5 bancos de dados menores (e menos carregados) no servidor, escolha a arquitetura do SuperServer e configure cada banco de dados em databases.conf;
- Servidores pequenos;
- Ambientes onde o *cache* compartilhado é mais desejável que a escalabilidade do *Super Classic*;

Classic Server

- Ambientes onde a estabilidade é a principal prioridade;
- Servidores com vários processadores e muita memória (verificar quanto cada processo utiliza em média e multiplicar pelo total de conexões);
- Bancos de dados grandes com centenas de usuários;

Super Classic

- Se você tem muitos bancos de dados no servidor (de 10 a 1000), e eles são mais ou menos iguais em termos de carga, e seu aplicativo é estável (ou seja, você nunca viu “encerramento anormal” no firebird.log), escolha SuperClassic. É melhor do que o Classic em termos de melhor controle de classificação de memória e desempenho;
- Servidores com vários processadores;
- Bancos de dados grandes com centenas de usuários;
- Ambientes onde o *cache* dedicado é mais vantajoso que o compartilhado do *Super Server*;
- Ambientes onde o *Classic Server* não se adapta mais;

Voltar à página inicial

Revision #11
Created Tue, May 7, 2019 4:51 PM
Updated Tue, Dec 15, 2020 6:02 PM