Nota Técnica: Alocação de Recursos e Configuração do MySQL para Otimização de Desempenho

Data: [Data]*

Autor: Engenheiro Jonatas Castelo Branco

Resumo:

Importância da Alocação de Arquivos de Dados e Logs em Discos Diferentes: Antes de discutirmos as recomendações prioritárias para otimização de desempenho, é fundamental compreender a estratégia de alocação de arquivos de dados e logs no MySQL. Alocar esses componentes em discos diferentes desempenha um papel crítico na melhoria do desempenho e na resiliência do sistema.

- Arquivos de Dados: Os arquivos de dados contêm registros e tabelas essenciais para o funcionamento do banco de dados. Alocar esses arquivos em um disco separado proporciona benefícios notáveis. Isso permite um acesso mais rápido e eficiente aos dados, reduzindo conflitos de I/O e melhorando o desempenho das consultas.

- Logs de Redo: Os logs de redo desempenham um papel vital na recuperação de dados em caso de falhas. Alocar esses logs em um disco dedicado garante que as operações de gravação possam ocorrer de maneira eficiente, minimizando os atrasos e garantindo a integridade dos dados.

- Logs Binários: Os logs binários registram todas as alterações no banco de dados. Tê-los em um disco separado é crucial para garantir a confiabilidade e a segurança das informações. Além disso, isso facilita processos de replicação e recuperação de dados.

Recomendações Prioritárias:

Antes de realizar quaisquer alterações nas variáveis de configuração, é altamente recomendável fazer um backup do arquivo de configuração `my.ini`. Isso é crucial para evitar transtornos com erros ou configurações incorretas. Lembre-se de que a configuração inadequada pode impactar negativamente o desempenho do MySQL.

Além disso, ao ajustar as variáveis de configuração, aconselhamos que você faça alterações uma de cada vez. Isso permitirá que você avalie o impacto de cada alteração e ajuste gradualmente as configurações para atender às necessidades específicas do seu ambiente.

Agora, passaremos a discutir as recomendações prioritárias para otimizar o desempenho do MySQL, incluindo alocação de memória, configuração de I/O e outros aspectos essenciais.

1. Tamanho do Buffer Pool:

- Ajustar o tamanho do buffer pool é fundamental para melhorar o desempenho do MySQL. Aloque uma quantidade adequada de memória para o buffer pool, que armazena dados frequentemente acessados em memória, reduzindo as operações de leitura de disco.

1.1 Verificar no WorkBech a quantidade de memória alocada para o MYSQL:

SELECT @@innodb_buffer_pool_size;

"innodb_buffer_pool_size=2G"

2. Tamanho dos Arquivos de Log (Log File Size):

- O tamanho dos arquivos de log, incluindo os logs de redo e binários, deve ser dimensionado adequadamente. Isso ajuda a evitar o enchimento prematuro dos logs e a garantir a disponibilidade contínua das informações de log.

3. Flush Log at Commit:

- Configure a opção "flush_log_at_commit" para equilibrar o desempenho e a durabilidade. Isso determina a frequência com que as operações de commit gravam no log de redo. Ajuste conforme necessário para atender aos requisitos de consistência e desempenho.

4. innodb_io_capacity:

- Defina o valor de `innodb_io_capacity` para uma estimativa da capacidade de E/S do seu sistema de armazenamento. Comece com um valor moderado, como 500 a 800, e ajuste com base no desempenho observado.

Ex: innodb_io_capacity = 2000

5. innodb_io_capacity_max:

- Configure `innodb_io_capacity_max` para um valor mais alto do que `innodb_io_capacity`, mas ainda assim, não muito alto para evitar sobrecarga. Um valor como 1000 a 1500 pode ser apropriado.

EX: innodb_io_capacity_max = 4000

6. sync_binlog:

- Configure `sync_binlog` para determinar a frequência com que os registros do log binário são sincronizados no disco. Um valor baixo, como `1`, garante alta durabilidade, mas pode afetar o desempenho. Escolha o valor com base nas necessidades de durabilidade do sistema.

7. table_open_cache:

- Configure `table_open_cache` para otimizar o uso de memória e melhorar o desempenho ao armazenar informações sobre tabelas abertas.

8. max_connections:

- Defina `max_connections` para limitar o número máximo de conexões simultâneas permitidas no servidor MySQL. Isso ajuda a evitar o esgotamento de recursos e a manter o desempenho do servidor em níveis aceitáveis.

9. Index Clustered e Non-Clustered

- Realizar análise de performace das query's para definir indices,

Clusted é definidor pela criação da primary key

Non-Clustered são definidos pelo user a partir da necessidade de otimização das buscas.

Ex - "CREATE INDEX idx_UsersNameColuna ON users(COLUNA);

OBS - Verificar se a coluna referenciada não se encontra em tipo TEXT, pois o MYSQL, caso esteja alterar para VARCHAR(255).

9.1 - Query para observar as estatísticas do índices, uma forma de observar a cardinalidade de cada tabela

"SHOW INDEX FROM USERS FROM STACKOVERFLOW;"

9.2 - A tabela innodb_table_stats inclue uma coluna que mostra quando as estatisticas foram atualizadas

"select * from mysql.innodb_table_stats;"

Query mostra todas as estatisticas dos indices e a cardinalidade de cada estatistica.

"select database_name,table_name,index_name,stat_name,stat_description

from mysql.innodb_index_stats where table_name like 'nome_da_tabela' and database_name like 'nome_do_banco';"

OBS - Após verificar as estatísticas, e constatado que estão desatualizadas por alguma motivo, force a atualização:

"Analyze table nome_da_tabela:"

9.3 - Query que mostra os processos que estão rodando no mysql

"show processlist"

Outras Recomendações:

1. Dados do Banco de Dados:

- É altamente recomendável alocar os dados do banco de dados em um dispositivo de armazenamento dedicado, separado dos logs de redo e binários. Isso proporciona um acesso mais rápido e eficiente aos dados, minimizando conflitos de I/O.

Benefícios:

A implementação das práticas recomendadas acima oferece diversos benefícios:

- Otimização do desempenho das operações de leitura e gravação de dados.

- Maior resiliência do sistema, reduzindo o risco de perda de dados em caso de falhas.

- Facilitação da manutenção e do backup de dados.

- Minimização de gargalos de I/O, especialmente em cargas de trabalho intensivas.

- Melhoria na eficiência da memória e no controle de conexões.

Conclusão:

A alocação eficiente de recursos e a configuração adequada no MySQL desempenham um papel crucial na manutenção de um ambiente de banco de dados de alto desempenho e confiabilidade. Implementar as práticas recomendadas mencionadas nesta nota técnica, com ênfase na alocação de recursos e configurações para "buffer pool size," "log file size," "flush log at commit," "innodb_io_capacity," "innodb_io_capacity_max," "sync_binlog," "table_open_cache," e "max_connections," ajudará os DBAs a otimizar o desempenho do sistema e a garantir a integridade dos dados.

Fim da Nota Técnica