Aprofundamento (Banco de Dados)

Certified Developer
The Ultimate Tech Degree

DigitalHouse >
Coding School

Temas

1. Otimização em Banco de dados.
2. Otimização e usuarios.
3. Outras tecnologias de banco de dados.

1 | Otimização em Consultas.

Otimização em Consultas

Ainda trabalhando com índices....

De forma simplificada, um índice reúne os valores de uma coluna em ordem alfabética (ou numérica) para que o MySQL possa encontrar um valor determinado mais rapidamente, em vez de procurar linha por linha.

SQL

```sql
CREATE INDEX nome_indice ON
table(atributo);
```

Otimização em Consultas

Função Explain

Aplicando o comando **EXPLAIN** antes do script *SELECT* retorna um registro contendo a análise do script para aquela tabela consultada.
Esse método retorna algumas informações:

```sql
EXPLAIN SELECT * FROM albuns;
```

Otimização em Consultas

Função Explain (resultado)

- **POSSIBLE_KEYS:** mostra os índices disponíveis para serem utilizados naquela consulta;
- **KEY:** mostra o índice escolhido pelo MySQL para realizar a consulta;
- **ROW:** número estimado de linhas percorridas para encontrar o resultado do SELECT;
- **EXTRA:** sugestões de otimização em índices da consultas, como usar DISTINCT, NOT EXIST, USING INDEX, entre outras.

Otimização em Consultas

Função Analyze

Use o comando **ANALYZE** para gerar uma distribuição de chaves, para a tabela a ser utilizada pelo otimizador de consultas do MySQL, para decidir quais índices serão melhores se utilizados em uma consulta.

```sql
ANALYZE TABLE albuns;
```

Otimização em Consultas

Função Priority

Se há mais consultas do que inserções de dados, você pode ter uma prioridade mais baixa do comando UPDATE ou SELECT, ou vice-versa, utilizando:

```sql
UPDATE LOW_PRIORITY...; //OU
SELECT HIGH_PRIORITY...;
```

Otimização em Consultas

Função Priority

Se o cliente não está interessado nos resultados do comando INSERT, então esta operação já pode imediatamente utilizar menos os recursos do sistema, utilizando-se o seguinte abaixo. **(Isso torna o sistema mais rápido porque acumula as inserções em lotes.)**

```sql
INSERT DELAYED;
```

Otimização em Consultas

Função Benchmark

Por fim, para saber quanto tempo uma determinada função ou expressão MySQL está demorando, use a função MySQL embutida:

```sql
SELECT BENCHMARK ( 10000, ( SELECT
col1 FROM funcionarios LIMIT 1 ) );
```

Lembrando que, para passar consultas na função BENCHMARK, obrigatoriamente a query deve retornar 1 linha com 1 coluna.

2 | Otimização e Permissões.

Otimização em Banco de dados

Permissões

Além de incrementar na segurança da conexão, criar usuários com permissões específicas reduzem o overhead das verificações das permissões em todos os comandos realizados no MySQL, melhorando assim no desempenho do sistema.

Criando e exibindo usuários:

```sql
CREATE USER ‘new_user’@’servidor’
IDENTIFIED BY ’password’;
SELECT user FROM mysql.user;
```

Otimização em Banco de dados

Permissões

Dando permissões ao usuário...

```sql
GRANT Oper1,...,OperN ON db_name.tb_name TO user_name@localhost;
```

Você pode pensar em estratégias de multiplos usuários, um para leitura, outra pra gravação e etc...

Otimização em Banco de dados

Permissões

Exibindo permissões do usuário...

```SQL
SHOW GRANTS FOR user_name@localhost;
```

Você pode pensar em estratégias de multiplos usuários, um para leitura, outra pra gravação e etc...

Otimização em Banco de dados

Permissões

Revogando permissões do usuário...

```SQL
REVOKE Operation ON db_name.tb_name FROM user_name@localhost;
```

Otimização em Banco de dados

Permissões

Excluindo um usuário...

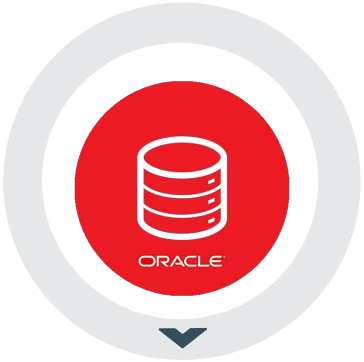
```sql
DROP USER user_name@localhost;
```

3 | Outras tecnologias em banco de dados.

Outros bancos relacionais...

SQL Server

- Banco Relacional
- Se destaca pela segurança e estabalidade.
- Criado nos anos 80.



Oracle DB

- Banco Relacional
- Se destaca pelo sistema de gerenciamento e pela padronização.
- Complexidade e incompatibilidade.

DB2 IBM

- Banco Relacional
- Facilidade e flexibilidade na implementação.
- Alta capacidade (alta compressão de dados).

Principais bancos de dados não relacionais (NoSQL).

MONGO DB

- Banco NoSQL (Documento)
- Escala elástica (melhor escalabilidade)
- Economia de recursos computacionais.

Cassandra DB

- Banco noSQL (Coluna)
- Ideal para grande volume de dados (usado pelo Facebook)
- Ideal para sistemas distribuidos.

Redis

- Banco noSQL (chave/valor)
- Compatibilidade com diversas linguagens.
- Versatilidade e facilidade

Comparação de bancos de dados NoSQL.

Conclusão

Nesta aula pudemos ver um pouco mais sobre funções que podem otimizar nosso banco de dados mySQL.

Aprendemos também sobre outros banco de dados e quando são viáveis para os projetos.


DigitalHouse>
Coding School