Índices

Certified Developer
The Ultimate Tech Degree

DigitalHouse >
Coding School

Índice

1 ÍNDICES

2 CREATE INDEX

3 DROP/ALTER INDEX

4 ANALYZE TABLE

1 Índices

Índices

Um índice dentro de um banco de dados é uma estrutura de dados que melhora a velocidade das consultas, por meio de um identificador único para cada linha de uma tabela. Tal, permite acesso rápido aos registros de uma tabela em um banco de dados.

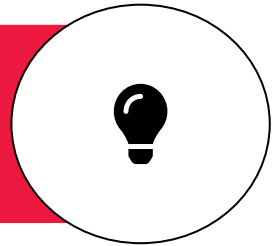
Mas, o que isso significa?

Índices

Vejamos um exemplo sobre uma tabela de alunos:

Id_Aluno (PK)	Nome	Sobrenome	Email	Data_de_Nascimento
1	Jose	Mentos	jmentos@gmail.com	29/5/2002
2	Laura	Gomez	laug93@gmail.com	02/3/1993
3	Lucas	Estevanez	estelu@gmail.com	31/3/2000

Se buscarmos um aluno pelo seu ID, seria fácil porque se trata de um elemento único, ordenado e estruturado.



Índices

O que acontece se um aplicativo precisar pesquisar alunos por e-mail?

Id_Aluno (PK)	Nome	Sobrenome	Email	Data_de_Nascimento
1	Jose	Mentos	jmentos@gmail.com	29/5/2002
2	Laura	Gomez	laug93@gmail.com	02/3/1993
3	Lucas	Estevanez	estelu@gmail.com	31/3/2000

Sempre que você precisar encontrar um aluno, terá que pesquisar célula por célula até encontrar o e-mail solicitado. Com 3 linhas é fácil, mas e se tivermos uma base de 1 milhão de alunos? E se a pesquisa for feita repetidamente durante o dia?

Índices

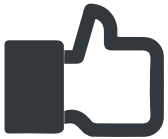
Teremos problemas de desempenho. Cada pesquisa que fizermos levará muito tempo para ser respondida.

Os índices são usados para resolver isso!

Um índice é uma estrutura adicional, onde escolhemos 1 ou mais colunas que farão parte do índice. Isso permitirá que você localize rapidamente os dados da tabela com base em seu conteúdo na (s) coluna (s) indexada (s).

Índices - vantagens

Embora pareça muito bom, o desempenho de um índice tem suas vantagens e desvantagens:



- O uso de índices pode melhorar o desempenho da consulta porque os dados necessários para satisfazer as necessidades da consulta existem no próprio índice.
- Eles podem melhorar significativamente o desempenho se a consulta contiver agregações (GROUP BY), junções de tabela (JOIN) ou uma mistura de agregações e junções.

Índices - desvantagens

- As tabelas usadas para armazenar os índices ocupam espaço.
- Os índices consomem recursos, pois cada vez que uma operação de atualização, inserção ou exclusão é realizada na tabela indexada, todas as tabelas de índice definidas nela devem ser atualizadas também (na atualização, apenas os índices definidos nela é necessário para a atualização das colunas).

Por essas razões, não é uma boa ideia definir índices indiscriminadamente.

Conclusão

- **Evite criar muitos índices** em tabelas que são atualizadas com muita frequência e tente defini-los com o mínimo de colunas possível.
- É conveniente usar um número maior de índices para melhorar o desempenho das consultas em tabelas com pouca necessidade de atualização, mas com grandes volumes de dados.
- É recomendável usar um comprimento de chave curto para índices agrupados. Os índices agrupados também são aprimorados se forem criados em colunas únicas ou sem valores (NULL).

Tipos de índices:

Simples: um índice simples é definido em uma única coluna.

```SQL
CREATE INDEX "I_LIVROS_AUTOR"
ON "LIVROS" (AUTORES);
```

Tipos de índices:

Composto: um índice composto é feito de várias colunas da mesma tabela

```sql
CREATE INDEX "I_LIVROS_AUTOREDITORIAL"
ON "LIVROS" (AUTOR,EDITORIAL);
```

Tipos de índices:

Índice agrupado: um índice agrupado (CLUSTERED) é um índice que armazena os dados nas linhas em ordem.
Apenas um único índice agrupado pode ser criado em uma tabela de banco de dados.
Isso funciona de forma eficiente apenas se os dados forem classificados em ordem crescente ou decrescente.

```SQL
CREATE CLUSTERED INDEX "I_LIVROS_AUTOR"
ON "LIVROS" (AUTOR);
```

Tipos de índices:

Índice não agrupado: os dados são organizados aleatoriamente, mas o índice especifica internamente uma ordem lógica. A ordem do índice não é a mesma que a ordem física dos dados.
Os índices não agrupados funcionam bem com tabelas em que os dados são modificados com frequência e o índice é construído nas colunas usadas na ordem pelas instruções WHERE e JOIN.

```SQL
CREATE NONCLUSTERED INDEX "I_LIVROS_AUTOR"
ON "LIVROS" (AUTOR);
```

2 | CREATE INDEX

Sintaxe CREATE INDEX

Com **CREATE INDEX** podemos criar um índice indicando as colunas envolvidas:

```sql
CREATE INDEX "NOME_ÍNDICE"
ON "NOME_TABELA" (NOME_COLUNA);
```

```sql
CREATE INDEX "I_LIVROS_AUTOREDITORIAL"
ON "LIVROS" (AUTOR,EDITORIAL);
```

3 | DROP/ALTER INDEX

Sintaxe DROP INDEX

Com **DROP INDEX**, podemos excluir um índice de uma determinada tabela:

```sql
DROP INDEX `nome_indice` ON `tabela1`;
```

4 | ANALYZE TABLE

Sintaxe ANALYZE TABLE

Com ANALYZE TABLE é possível analisar e armazenar a distribuição de chaves para uma tabela:

```sql
ANALYZE TABLE "NOME_TABELA";
```

Sintaxe EXPLAIN

Com EXPLAIN é possível obter mais informações sobre a consulta. Assim como o índice utilizado na consulta.

```sql
EXPLAIN "NOME_TABELA";
```


DigitalHouse>
Coding School