Índices no PostgreSQL

1

Índices

- Os índices são um modo comum de melhorar o desempenho do banco de dados.
- O índice permite ao servidor de banco de dados encontrar e trazer linhas específicas muito mais rápido do que faria sem o índice.
- O índice aponta onde está um determinado registro, ou seja, ele é um direcionador de modo que a consulta seja rápida, precisa e portanto eficiente.
- Entretanto, os índices também produzem trabalho adicional para o sistema de banco de dados como um todo devendo, portanto, serem utilizados com sensatez.

2

2

Índice

- Caso a tabela não tenha índice será necessária a verificação registro por registro – seq scan – até encontrar o que você procura;
- Esse processo demanda muito processamento e pode, inclusive, derrubar o servidor.

Tipos de índice

- O PostgreSQL disponibiliza vários tipos de índice: B-tree (árvore B), hash, GiST e GIN.
- Cada tipo de índice utiliza um algoritmo diferente, mais apropriado para tipos diferentes de consulta.
- Por padrão, o comando CREATE INDEX cria um índice B-tree, adequado para a maioria das situações comuns.

Índice B-tree

- Podem tratar consultas de igualdade e de faixa, em dados que podem ser classificados em alguma ordem.
- Indicado para consultas com os operadores:
  - <, <=, =, >=, >.
  - Também pode ser utilizado com LIKE, ILIKE, ~ e ~*
- Exemplo:

  CREATE INDEX nome ON tabela USING BTREE (coluna);

5

5

Índice R-tree

- Usados em consultas a dados espaciais.
- Adequado para consultas com os operadores:
  - <<, &<, &>, >>, @, ~=, &&.
- Não existe nas versões 9.1 e superiores;
- Foi substituído pelo GIST;
- Exemplo:

  CREATE INDEX nome ON tabela USING RTREE (coluna);

6

6

Índice Hash

- Indicados para consultas com comparações de igualdade simples.
- É desencorajado seu uso.
- Em seu lugar recomenda-se o B-tree.
- Testes mostraram que os ínides hash não têm desempenho melhor que os índices B-tree, e que o tamanho e o tempo de construção dos índices hash são muito piores.

- Exemplo:

```
CREATE INDEX nome ON tabela USING HASH (coluna);
```


Índice GiST

- Não são um único tipo de índice, mas uma infraestrutura dentro da qual podem sem implementadas estratégias de indexação diferentes.
- Por exemplo, o GiST suporta classes de operadores para vários tipos de dados geométricos bidimensionais:
  - <<, &<, &>, >>, <<|, &<|, |&>, |>>, @>, <@, ~=, &&

- Os operadores em particular com os quais o índice GiST pode ser utilizado variam dependendo da estratégia de indexação (a *classe de operadores*):

Índice GIN

- São índices invertidos que podem manipular valores que contêm mais de uma chave, tais como arrays.
- Assim como o GiST, GIN pode suportar várias estratégias diferentes de indexação definidas pelo usuário
- Por exemplo, o GIN inclui operadores de classes para arrays unidimensionais, que suportam consultas indexadas usando os operadores:
  - @>, <@, =, &&

Exemplos de Índices

- Criando um índice:
  - CREATE INDEX idx_pessoas_nome ON pessoas USING hash (nome);
- Criando um índice que “incorpora” dos campos ao mesmo tempo:
  - CREATE INDEX idx_usuario_login_senha ON usuario USING btree (login,senha);
- Criando um índice, mas indexando apenas parte do campo:
  - CREATE INDEX idx_pessoas_endereco ON pessoas USING btree (substr(endereco, 1, 10));

Exemplos de Índices

- Criando um índice em um determinado tablespace:
  - CREATE INDEX idx_compras ON compras (item) TABLESPACE outro_tablespace;
- Criando um índice com a cláusula WHERE:
  - CREATE UNIQUE INDEX ind_nome_tiago ON pessoa USING GIN (nome) WHERE nome iLIKE 'Tiago%';
- Renomeando um índice:
  - CREATE INDEX idx_pessoas_endereco RENAME TO idx_pessoas_endereco_completo;
- Excluindo (dropando) um índice:
  - DROP INDEX idx_pessoas_endereco_completo

Outros Atributos de um Índice

- UNIQUE:
  - Não irá indexar registros duplicados e retornará um erro.
  - Exemplo:
    - CREATE INDEX UNIQUE idx_pessoa_cpf ON pessoas USING btree (cpf);
- CONCURRENTLY:
  - Caso seja declarado ele informa que não será realizado o bloqueio do índice durante procedimentos de DML (insert, update, etc).
  - Exemplo:
    - CREATE INDEX CONCURRENTLY idx_pessoa_rg ON pessoas USING btree (rg);