INDICI
A CHE CAZZO SERVONO E COME USARLI

Un esempio pratico:

Immaginiamo di avere una tabella chiamata utenti che contiene informazioni sugli utenti di un sistema. La tabella ha le seguenti colonne:

id: identificatore univoco dell'utente

nome: nome dell'utente

email: email dell'utente

data_di_registrazione: data di registrazione dell'utente

- **Senza Indici**

Quando esegui una query per cercare un utente specifico per email, senza indici, il database deve scansionare ogni riga della tabella per trovare quella che corrisponde all'email specificata. Questo processo è noto come scansione completa della tabella.

sql

*SELECT * FROM utenti WHERE email = 'esempio@dominio.com';*

Se la tabella contiene molti utenti, questa operazione può richiedere molto tempo.

- **Con Indici**

Ora vediamo come un indice può migliorare le prestazioni di questa query. Creiamo un indice sulla colonna email.

sql

CREATE INDEX idx_utenti_email ON utenti (email);

Con questo indice, PostgreSQL crea una struttura dati (simile a un indice in un libro) che associa ogni email alla riga corrispondente. Quando esegui la stessa query di prima, PostgreSQL può utilizzare l'indice per trovare rapidamente la riga corrispondente senza dover scansionare ogni riga della tabella.

Esempio di Indice B-tree

Un indice B-tree è una struttura ad albero bilanciata che consente una ricerca efficiente. Ecco come creare un indice B-tree sulla colonna email della tabella utenti.

sql

CREATE INDEX idx_utenti_email ON utenti (email);

Quando esegui una query che cerca un utente per email, PostgreSQL utilizza l'indice B-tree per navigare rapidamente attraverso la struttura ad albero e trovare l'email specificata.

Vantaggi dell'Indice

- Velocità di Ricerca: L'indice permette di trovare rapidamente l'email senza dover scansionare ogni riga della tabella.
- Prestazioni: Migliora significativamente le prestazioni delle query di ricerca.

Visualizzazione degli Indici

Per visualizzare gli indici creati su una tabella, puoi usare il comando \d seguito dal nome della tabella in psql, il client interattivo di PostgreSQL.

\d utenti

Questo comando mostrerà le colonne della tabella utenti e gli indici associati.

DEFINIZIONI E TIPOLOGIE :

Gli indici in un database sono strutture di dati che migliorano la velocità delle operazioni di recupero dei dati. Funzionano come un indice in un libro, permettendo di trovare rapidamente le informazioni senza dover scorrere tutte le pagine. Nel contesto di PostgreSQL, gli indici sono essenziali per ottimizzare le query, riducendo il tempo necessario per eseguire ricerche, ordinamenti e join.

Tipi di Indici in PostgreSQL: B-tree e Hash

- **Indice B-tree:**

È l'indice predefinito in PostgreSQL. Ottimale per una vasta gamma di query. Supporta uguaglianze (=), range (<, <=, >, >=), e pattern matching con LIKE che non inizia con un wildcard. Adatto per colonne con distribuzione di dati uniforme e range queries.

- **Indice Hash:**

Specifico per uguaglianze (=). Non supporta range queries. Può essere più veloce degli indici B-tree per operazioni di uguaglianza pura, ma ha un uso limitato.

Decisione del Tipo di Indice da Utilizzare

La scelta tra indici B-tree e Hash dipende dal tipo di query che si prevede di eseguire sul database:

- B-tree è più versatile e generalmente preferibile a meno che non si abbiano specifiche necessità di uguaglianza pura.
- Hash può essere vantaggioso per colonne che vengono utilizzate esclusivamente con operazioni di uguaglianza.

Analisi del Progetto "Gestione di un Museo d'Arte"

Dopo aver esaminato il documento, possiamo identificare le aree chiave dove l'uso degli indici può migliorare le prestazioni. Ecco alcune considerazioni e proposte:

- **Entità DIPENDENTE:**

Campi importanti: CF (Codice Fiscale), Nome, Cognome, Email.

Le query probabilmente cercheranno dipendenti per CF e Email (unici), quindi un indice B-tree su questi campi è essenziale.

- **Entità OPERA:**

Campi importanti: Codice, Autore, Titolo, Data di Produzione.

Le query potrebbero cercare opere per Codice (unico), Autore, e range di Data di Produzione.

- **Entità MOSTRA:**

Campi importanti: Nome, Curatore, Data di Inizio, Data di Fine.

Le query potrebbero includere ricerca per nome della mostra e range di date.

- **Entità VISITATORE:**

Campi importanti: Email, Nome, Cognome.

Le query cercheranno visitatori per Email (unico), quindi un indice B-tree su questo campo è essenziale.

\

Implementazione degli Indici in PostgreSQL :

```
/*Indici per opere interne */
CREATE INDEX idx_operaI_codice ON operainterna (id);
CREATE INDEX idx_operaI_autore ON operainterna (id_artista);
CREATE INDEX idx_operaI_data_produzione ON operainterna ( AnnoProduzione);

/*Indici per opere esterne */
CREATE INDEX idx_operaE_codice ON operaesterna (id);
CREATE INDEX idx_operaE_autore ON operaesterna (id_artista);
CREATE INDEX idx_operaE_data_produzione ON operaesterna  (AnnoProduzione);


/*Indici per i dipendenti */
CREATE INDEX idx_Direttore_cf ON Direttore (cf);
CREATE INDEX idx_Direttore_email ON Direttore (email);

CREATE INDEX idx_curatore_cf ON curatore (cf);
CREATE INDEX idx_curatore_email ON curatore (email);

CREATE INDEX idx_Restauratore_cf ON Restauratore (cf);
CREATE INDEX idx_Restauratoree_email ON Restauratore (email);
```

```sql
CREATE INDEX idx_Registrar_cf ON Registrar (cf);

CREATE INDEX idx_Registrar_email ON Registrar (email);
```

Motivazione delle Scelte

- Indice B-tree su cf e email di dipendente:
    - Motivazione: Permette rapide ricerche e garantisce unicità.
- Indice B-tree su codice, autore e data_produzione di opera:
    - Motivazione: codice è unico, mentre autore e data_produzione sono utilizzati per ricerche frequenti e ordinamenti.
- Indice B-tree su nome, curatore, data_inizio e data_fine di mostra:
    - Motivazione: Facilita le ricerche per nome di mostra e permette efficienti range queries sulle date delle mostre.
- Indice B-tree su email di visitatore:
    - Motivazione: Permette identificazione univoca e ricerche rapide per email.

NB: Gli indici in PostgreSQL vengono utilizzati automaticamente dal database per ottimizzare le query. Non è necessario specificare esplicitamente l'uso di un indice nelle query SQL; PostgreSQL decide automaticamente quando utilizzare un indice basandosi sul piano di esecuzione della query. Tuttavia, si puo verificare se un indice viene utilizzato tramite il comando EXPLAIN.