Index

**Cấu trúc của Index trong Database gồm có hai cột:**

● Cột Search Key: Chứa bản sao các giá trị của cột được tạo index. Đây là giá trị mà người dùng thường tìm kiếm.

● Cột Data Reference: Chứa con trỏ đến địa chỉ của bản ghi có giá trị cột index tương ứng. Điều này cho phép hệ quản trị cơ sở dữ liệu (DBMS) nhanh chóng xác định vị trí của dữ liệu thực tế trong bảng

**Clustered index thường được sử dụng cho các cột có giá trị duy nhất và thường xuyên được truy cập.**

**Non-clustered index thường được sử dụng cho các cột không phải là khóa chính và thường xuyên được truy cập trong các truy vấn tìm kiếm, sắp xếp hoặc thống kê dữ liệu.**

1. Phân biệt cluester index và non-cluster index ? So sánh về perfomance khi sử dụng 2 loại index này ? Giải thích nguyên nhân ?

Một **Clustered Index** sắp xếp dữ liệu vật lý của bảng theo một thứ tự duy nhất dựa trên các giá trị của cột chỉ mục. Hãy hình dung nó giống như một cuốn từ điển: các từ được sắp xếp theo thứ tự bảng chữ cái, và các định nghĩa (dữ liệu) được lưu trữ ngay cạnh chúng.

* **Lưu trữ dữ liệu:** Các hàng dữ liệu thực tế được lưu trữ theo thứ tự của chỉ mục.
* **Số lượng:** Mỗi bảng chỉ có thể có **một** Clustered Index.
* **Hoạt động:** Khi bạn tìm kiếm, cơ sở dữ liệu sẽ nhanh chóng đi đến vị trí của bản ghi và có thể đọc các bản ghi liền kề một cách hiệu quả.

**create** **unique** **index** idx\_employee\_id **on** employees(emp\_id)

**cluster** employees **using** idx\_employee\_id;

Một **Non-Clustered Index** không sắp xếp dữ liệu vật lý. Nó là một cấu trúc dữ liệu riêng biệt, giống như mục lục của một cuốn sách. Nó chứa một bản sao của các cột được chỉ mục và một **con trỏ** trỏ đến vị trí vật lý của hàng dữ liệu tương ứng.

* **Lưu trữ dữ liệu:** Dữ liệu vật lý của bảng không thay đổi thứ tự. Chỉ mục là một cấu trúc riêng biệt.
* **Số lượng:** Một bảng có thể có **nhiều** Non-Clustered Index.
* **Hoạt động:** Để tìm một bản ghi, cơ sở dữ liệu phải tìm trong chỉ mục, sau đó dùng con trỏ để đi đến vị trí dữ liệu thực tế.

|  |  |  |
| --- | --- | --- |
| Đặc điểm | Cluster Index | Non-cluster Index |
| Truy vấn toàn bộ bản ghi | **Rất nhanh.** Dữ liệu đã được sắp xếp, nên việc tìm kiếm và lấy dữ liệu liền mạch. | **Chậm hơn.** Phải thực hiện 2 bước: tìm trong chỉ mục, sau đó đi đến vị trí dữ liệu thực tế. |
| Truy vấn phạm vi | **Cực kỳ hiệu quả.** Khi tìm kiếm một dải giá trị, cơ sở dữ liệu chỉ cần tìm đến giá trị đầu tiên trong dải và sau đó đọc các hàng liền kề trên đĩa. | **Kém hiệu quả hơn.** Cơ sở dữ liệu phải thực hiện nhiều lần "tìm kiếm + đi đến vị trí" vì dữ liệu không được sắp xếp liền kề. |
| Thao tác Insert/Update | **Có thể chậm.** Khi chèn một bản ghi mới, cơ sở dữ liệu phải tìm đúng vị trí và có thể phải di chuyển các hàng khác để duy trì thứ tự vật lý. | **Có thể nhanh hơn.** Không cần phải sắp xếp lại dữ liệu vật lý. Chỉ cần cập nhật cấu trúc chỉ mục. |
| Dung lượng lưu trữ | **Ít tốn kém.** Không cần lưu trữ bản sao của toàn bộ các hàng dữ liệu, chỉ lưu trữ chỉ mục. | **Tốn kém hơn.** Cần thêm dung lượng để lưu trữ cấu trúc chỉ mục riêng biệt. |

**Nguyên nhân sự khác biệt về hiệu suất**

Nguyên nhân cốt lõi là sự khác biệt về cách lưu trữ dữ liệu.

* **Clustered Index:** Vì dữ liệu thực sự được sắp xếp theo chỉ mục, việc đọc dữ liệu theo thứ tự hoặc theo một phạm vi là **"đọc tuần tự"** trên đĩa. Điều này rất nhanh vì nó giảm thiểu các thao tác truy cập ngẫu nhiên trên đĩa.
* **Non-Clustered Index:** Nó chỉ là một bản đồ (map). Mỗi lần tìm kiếm, cơ sở dữ liệu phải thực hiện một thao tác truy cập ngẫu nhiên để đi từ chỉ mục đến vị trí của dữ liệu. Điều này đặc biệt kém hiệu quả khi bạn cần lấy nhiều hàng dữ liệu không nằm liền kề nhau trên đĩa.

2. Có giới hạn việc đánh bao nhiêu index trong 1 table hay không ? Giải thích ?

**Giới hạn thực tế và Nguyên nhân**

Việc tạo quá nhiều index sẽ gây ra các vấn đề nghiêm trọng về hiệu suất.

1. **Chi phí lưu trữ**: Mỗi index là một cấu trúc dữ liệu riêng biệt. Càng nhiều index, dung lượng lưu trữ cần thiết càng lớn, làm tăng kích thước của cơ sở dữ liệu.
2. **Hiệu suất ghi dữ liệu (INSERT, UPDATE, DELETE)**: Đây là tác động lớn nhất. Khi bạn thêm, sửa hoặc xóa một bản ghi, cơ sở dữ liệu phải cập nhật tất cả các chỉ mục liên quan. Càng nhiều index, chi phí cập nhật càng cao, làm các thao tác ghi dữ liệu trở nên chậm hơn đáng kể.
3. **Chi phí tối ưu hóa truy vấn**: Khi có quá nhiều index, trình tối ưu hóa truy vấn (query optimizer) phải mất nhiều thời gian hơn để phân tích và chọn chỉ mục tốt nhất để sử dụng cho một truy vấn. Trong một số trường hợp, nó có thể chọn sai chỉ mục, làm giảm hiệu suất.

3. Việc đánh index dựa trên cơ sở nào ?

**Tính chọn lọc cao (High Selectivity):** Chọn các cột có nhiều giá trị duy nhất (unique values) hoặc ít giá trị trùng lặp. Càng nhiều giá trị duy nhất, index càng hiệu quả trong việc giúp cơ sở dữ liệu tìm kiếm nhanh chóng.

**Ví dụ:** Các cột như email, user\_id, product\_code thường có tính chọn lọc cao và là ứng cử viên tốt cho việc đánh index. Ngược lại, cột gender chỉ có hai giá trị Nam/Nữ, nên việc đánh index trên cột này không mang lại nhiều lợi ích.

**Được sử dụng trong các mệnh đề truy vấn:** Index sẽ phát huy tác dụng tốt nhất khi được đánh trên các cột thường xuyên xuất hiện trong các mệnh đề WHERE, JOIN, và ORDER BY.

* WHERE: Lọc dữ liệu.
* JOIN: Nối dữ liệu giữa các bảng.
* ORDER BY: Sắp xếp dữ liệu.

**Thường xuyên được truy vấn:** Nếu một cột ít khi được sử dụng trong truy vấn, việc đánh index lên nó có thể không cần thiết và thậm chí còn lãng phí tài nguyên.

4. Làm thế nào để biết 1 câu query đã sử dụng index hay chưa ?

**Sử dụng EXPLAIN, EXPLAIN ANALYZE**

EXPLAIN ANALYZE SELECT \* FROM employees WHERE hire\_date > '2024-01-01';