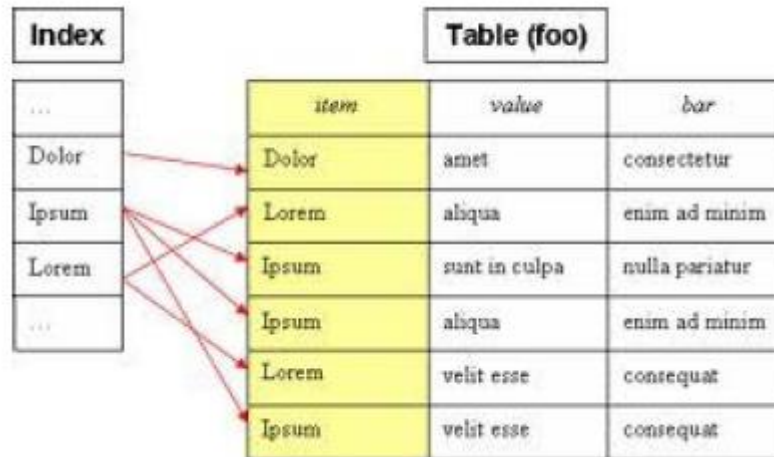


BÀI 6

CHỈ MỤC





Nội dung

1. Chỉ mục

1. Giới thiệu chỉ mục
2. Fill Factor

2. Các kiểu chỉ mục

1. Cluster Index
2. Non-Cluster Index
3. Đặc điểm của Index
4. Sử dụng và xóa Index



Chỉ mục INDEX

Cơ bản về chỉ mục (Index)

- Chỉ mục thường gắn kết với 1 bảng hay view để tăng tốc việc tìm kiếm hay khôi phục các hàng trong bảng hay view đó.
- Chỉ mục chứa các khóa (key) được trích từ 1 hay nhiều cột trong bảng dữ liệu hay view. Các khóa này được xếp thứ tự trong cấu trúc B-tree nhờ đó giúp SQL Server tìm kiếm nhanh chóng và hiệu quả các hàng trong bảng dữ liệu thông qua các giá trị khóa của index.



Chỉ mục INDEX

Cơ bản về chỉ mục

- Chỉ mục được tạo ra dựa theo các giá trị được xếp thứ tự từ 1 hay nhiều cột được chọn.
 - Chỉ mục được tạo tự động bất cứ lúc nào ta xác định khoá chính hay ràng buộc unique.
 - Có hai loại chỉ mục:
 - Clustered
 - Non - Clustered
- Cả hai đều là B-tree index
- Mỗi bảng chỉ có thể có duy nhất 1 chỉ mục clustered nhưng không bắt buộc là phải có chỉ mục này.
 - Một bảng có thể có tới 249 chỉ mục nonclustered



Chỉ mục INDEX

Mục đích sử dụng chỉ mục

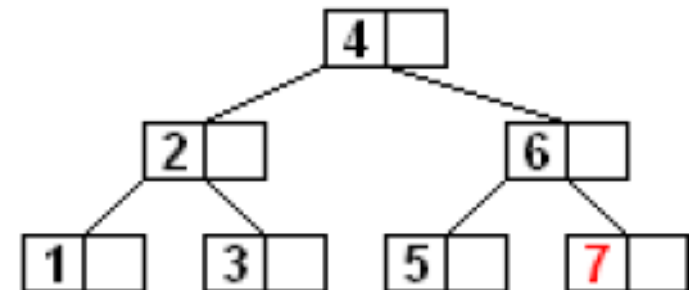
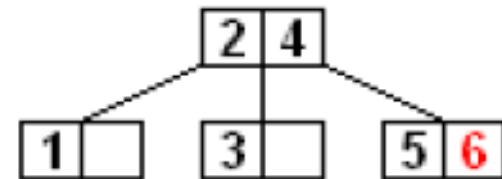
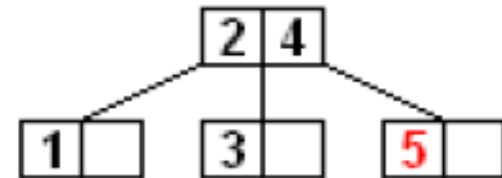
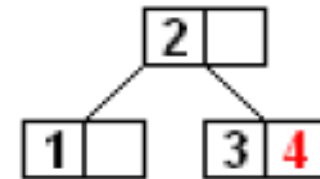
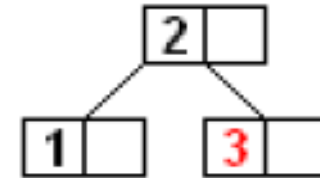
- Cải thiện việc thực thi khi sắp xếp hay nhóm dữ liệu
- Cải thiện việc thực thi các truy vấn có kết nối giữa các bảng
- Cải thiện tính duy nhất của 1 cột hay nhiều cột

Chỉ mục INDEX

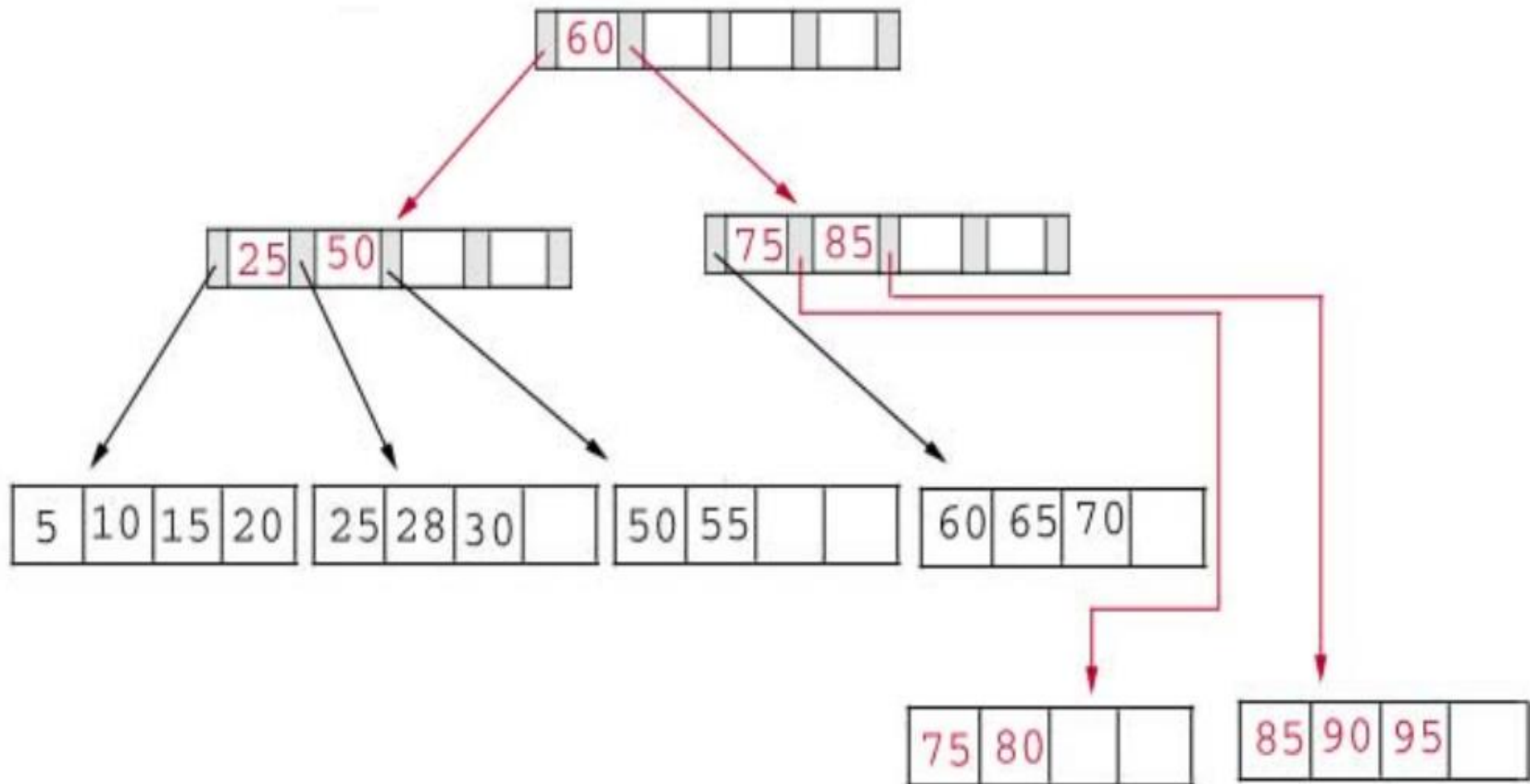
- Nếu bảng không dùng chỉ mục:
 - Các hàng không lưu trữ theo 1 thứ tự đặc biệt nào.
 - Các trang dữ liệu cũng không sắp xếp tuần tự.

1

1 2



Chỉ mục INDEX



Chỉ mục INDEX

Khóa Index được tạo từ cột emp_id của bảng employees.

employees table

emp_id	fname	minit	lname	job_id	job_lvl	...
PMA42628M	Paolo	M	Accorti			
PSA89086M	Pedro	S	Afonso			
VPS30890F	Victoria	P	Ashworth			
H-B39728F	Helen		Bennett			
L-B31947F	Lesley		Brown			
⋮	⋮	⋮	⋮			

Khi tìm kiếm một giá trị trong cột dữ liệu, mà cột này tham gia tạo khóa

Index, đầu tiên câu lệnh xác định vị trí của giá trị nằm trong khóa Index bằng phép duyệt cây, sau đó thực hiện tìm theo liên kết đến bản ghi chứa giá trị tương ứng với khóa trong bảng.

index on emp_id

⋮
L-B31947F
PMA42628M
⋮
VPA30890F
⋮



Chỉ mục INDEX

- Việc thiết kế khóa Index dựa trên nhu cầu truy vấn, chèn dữ liệu
- Cột thường được sử dụng làm khóa truy vấn dữ liệu (xác định cột tham gia bảng, xác định dựa vào một số tham số sau: khóa Index).
- Tập lệnh thường sử dụng truy vấn cần tốc độ cao (xác định tập cột tham gia truy vấn).
- Dữ liệu nhập vào bảng có khóa Index cần nhanh hơn hay truy vấn cần nhanh hơn (xác định đặt clustered hoặc nonclustered).
- Lượng dữ liệu nhập đồng loạt nhiều hay ít (xác định tham số fillfactor).



B-tree INDEX

- B-tree index nằm riêng trên những trang index, có 1 mức gốc (root level), một hay nhiều mức trung gian (intermediate levels), và 1 mức lá (leaf) hay mức node.
- Các cột được sắp xếp bởi b-tree index được gọi là cột (key) của index.
- Sự khác nhau giữa chỉ mục clustered và non-clustered là số lượng và loại dữ liệu được lưu trữ ở mức lá.



Chỉ mục INDEX

Clustered indexes

- Clustered indexes sắp xếp và lưu trữ các hàng dữ liệu trong bảng hay view theo giá trị khóa của index.
- Mỗi bảng chỉ có duy nhất 1 clustered index
- Hình ảnh tượng trưng của clustered index là telephone book
- Khi bảng có chứa clustered index thì các hàng của bảng được xếp thứ tự. Bảng còn được gọi là clustered table
- Nếu bảng không chứa clustered index thì các hàng của bảng lưu trữ tự do và được gọi là heap.



Chỉ mục INDEX

Clustered indexes

- Khi khóa đặt thuộc tính Clustered, dữ liệu của bảng sẽ được sắp xếp vật lý trên đĩa, như vậy khi thiết kế khóa dạng này dữ liệu được chèn và sẽ tìm đúng vị trí trên đĩa để lưu trữ (vùng đĩa dành cho bảng dữ liệu), chính vì vậy mà có thể xảy ra trường hợp phải dịch chuyển danh sách các giá trị đã có ở đĩa.
- Những việc tạo khóa Index dạng này sẽ không cần sắp xếp giá trị ở dạng logic mà khi truy nhập đĩa đã bảo đảm dữ liệu được sắp xếp.

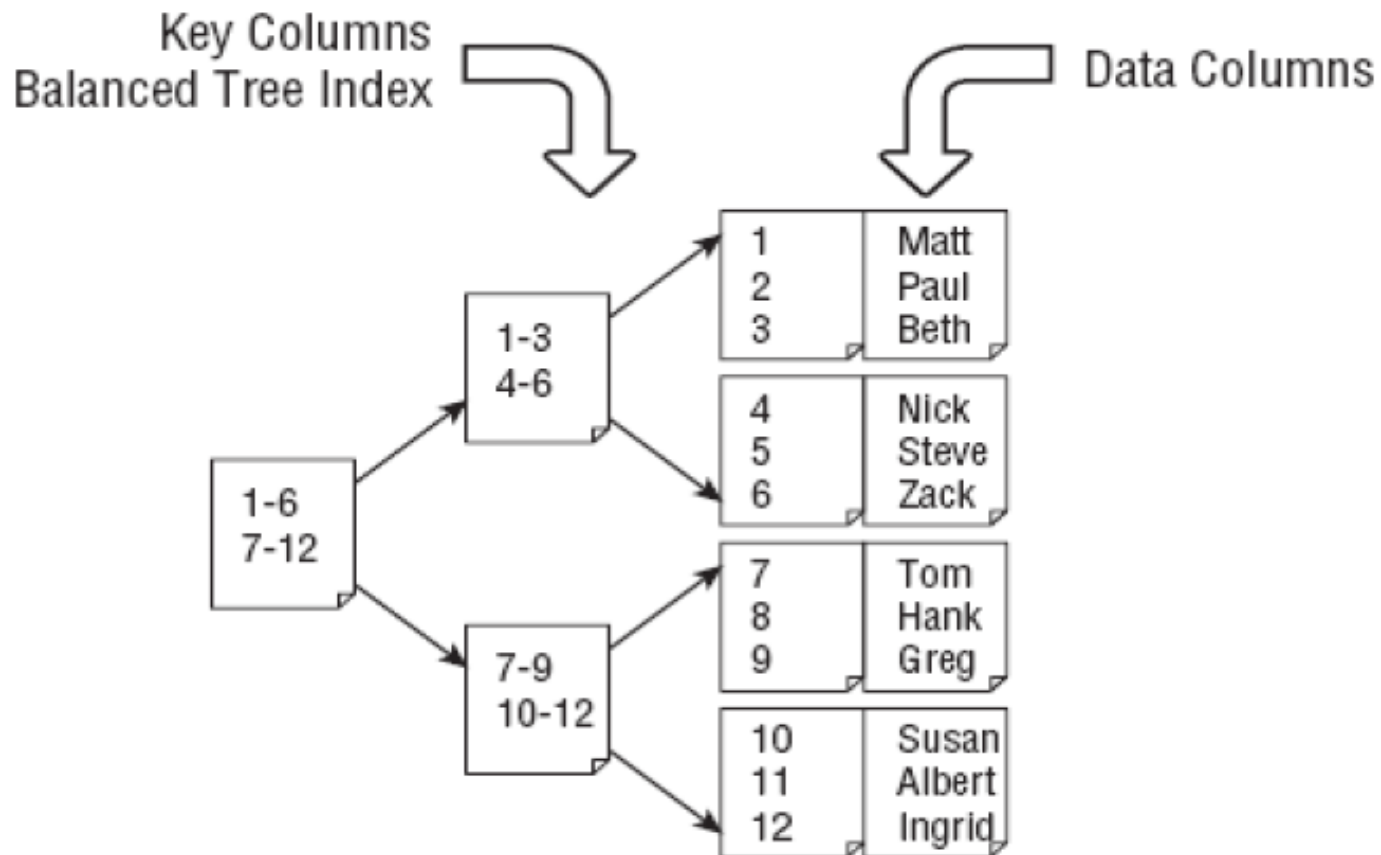


Chỉ mục INDEX

Clustered indexes

- Cũng có thể tạo 1 bảng không cần có Clustered indexes → dữ liệu được lưu trữ vào 1 heap không xếp thứ tự. Mỗi hàng sẽ xác định bởi mã RowID của Heap.
- RowID là vị trí vật lý thực sự của hàng, gồm 3 giá trị: FieldID:PageNum:SlotNum, và không thể truy vấn trực tiếp đến nó được.
- Chỉ mục non-clustered indexes lưu trữ RowID của Heap thay vì lưu trữ khóa chỉ mục clustered.

Chỉ mục INDEX



with an identity column as the clustered index key. The first name is the data column.



Chỉ mục INDEX

NonClustered indexes

- Dữ liệu Index không sắp xếp ở dạng vật lý mà chỉ sắp xếp logic, dữ liệu của bảng lưu trữ giá trị khóa Index được sắp xếp, nhanh trong nhập dữ liệu.
- Chỉ mục nonclustered tách riêng khỏi bảng dữ liệu.
- Chỉ mục nonclustered chứa các giá trị khóa và mỗi giá trị khóa có 1 con trỏ (ponter) trỏ đến hàng dữ liệu chứa giá trị khóa đó. Con trỏ này được gọi là row locator.
- Trong SQL Server 2008, một bảng có thể có tới 999 nonclutered index.

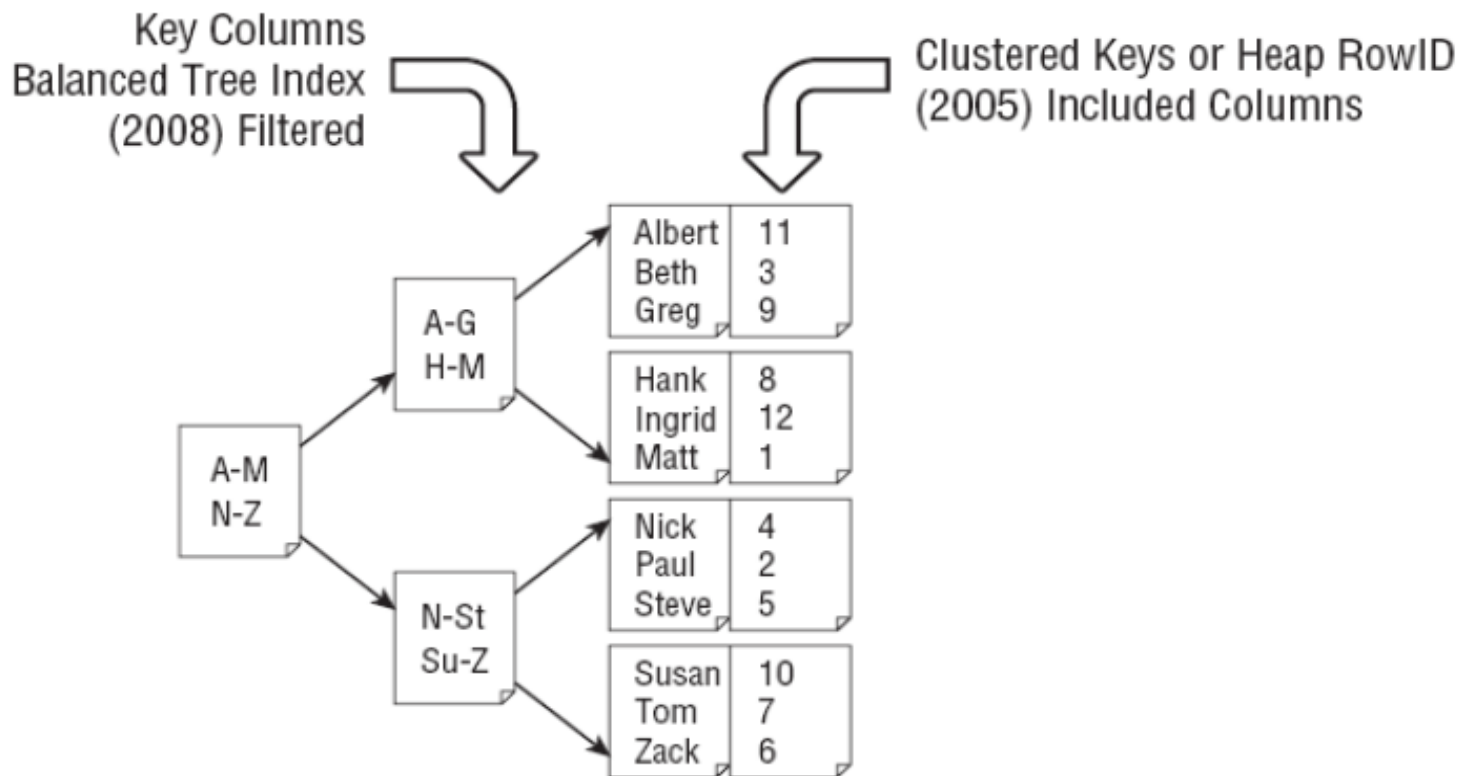


Chỉ mục INDEX

NonClustered indexes

- Cấu trúc của row locator phụ thuộc vào các trang dữ liệu được lưu trữ trong heap hay trong bảng clustered.
- Nếu trong heap, row locator là 1 con trỏ trỏ đến rowID của Heap
- Nếu trong bảng clustered, row locator là khóa chỉ mục clustered.
- Hình ảnh tượng trưng của nonclustered index là bảng chỉ mục nằm cuối sách.

Chỉ mục INDEX



first name as the key column. The non-clustered index includes pointers to the clustered index key column.



Chỉ mục INDEX

Cách tạo chỉ mục

```
CREATE [UNIQUE] [CLUSTERED] [NONCLUSTERED]  
INDEX Index_name ON Table (Column_name)  
[WITH  
    [PAD_INDEX]  
    [FILLFACTOR = fillfactor]  
    [[,] DROP_EXISTING]]
```



Chỉ mục Clustered và Nonclustered

- Chỉ mục *clustered*:
 - Dữ liệu được sắp xếp vật lý.
 - Chỉ có 1 chỉ mục clustered trong mỗi bảng.
 - Tương tự như danh bạ điện thoại (telephone directory) trong đó dữ liệu được sắp xếp bởi tên thuê bao

Ví dụ

```
CREATE CLUSTERED INDEX Customerid_ndx  
ON Orders (Customerid)
```



Chỉ mục Clustered và Nonclustered

- Chỉ mục *nonclustered*: là chỉ mục có cấu trúc riêng biệt độc lập với thứ tự vật lý của bảng dữ liệu.
 - Thứ tự vật lý của chỉ mục nonclustered không trùng với thứ tự các bản ghi trong bảng dữ liệu
 - Tương tự như chỉ mục trong textbook
- Nên tạo chỉ mục clustered trước khi tạo chỉ mục nonclustered để chỉ mục nonclustered không cần phải tạo lại.
- Chỉ mục clustered thực thi nhanh hơn chỉ mục nonclustered.

```
CREATE NONCLUSTERED INDEX Manv_ndx  
ON Nhanvien (Manv)
```



Xem - Xóa chỉ mục

Xem chỉ mục

Cú pháp

```
sp_helpindex <table_name>
```

Example

```
sp_helpindex Customers
```

Xóa chỉ mục

Cú pháp

```
sp_helpindex <table_name>
```

Example

```
sp_helpindex Customers
```



Chỉ mục INDEX

Unique Index

- Xác định dữ liệu của cột tham gia khóa Index không lặp lại.

Fill Factor

- Khi tạo khóa Index, dữ liệu tham giá tạo khóa Index sẽ được phân theo mức của B-Tree, các mức được phân theo page dữ liệu, giá trị Fill factor xác định phần khoảng trống tối đa của page theo tỷ lệ phần trăm. Nhờ khoảng trống này mà tốc độ bố trí cấu trúc Index, tốc độ truy lục thông tin trong cây được cải thiện.

Chỉ mục Composite

- Cho phép hai hay nhiều cột được sử dụng để tạo chỉ mục

Creating Unique Indexes



```
USE library
CREATE UNIQUE INDEX title_ident
ON title (title_no)
```

<i>title</i>			
<i>title_no</i>	<i>title</i>	<i>author</i>	<i>synopsis</i>
10	The Night-Born	Jack London	~ ~ ~
11	Lemon	Motojirou	~ ~ ~
12	Walking	Henry David Thoreau	~ ~ ~

*Duplicate key values are not allowed
when a new row is added to the table*

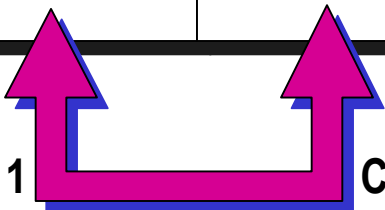


12	Le Petit Prince	Antoine de Saint-Exupery	~ ~ ~
----	-----------------	--------------------------	-------

Creating Composite Indexes

```
USE library
CREATE UNIQUE INDEX loan_ident
ON loan (isbn, copy_no)
```

<i>loan</i>				
<i>isbn</i>	<i>copy_no</i>	<i>title_no</i>	<i>member_no</i>	<i>out_date</i>
342	5	35	3744	1998-01-06
342	10	35	5278	1998-01-04
343	4	35	3445	1998-01-04

Column 1  Column 2

Composite Key



INDEX và truy vấn tối ưu

- Các Index được thiết kế tốt sẽ giảm thao tác I/O đĩa và tiêu tốn tài nguyên ít hơn.
- Các chỉ mục có thể hỗ trợ cho nhiều loại truy vấn chứa các lệnh SELECT, UPDATE, DELETE và MERGE.
- Khi thực hiện truy vấn, query optimizer đánh giá và chọn phương pháp nào hữu hiệu nhất để khôi phục dữ liệu
- Duyệt qua toàn bộ bảng (table scan)
- Duyệt qua một hay nhiều chỉ mục nếu có



INDEX và truy vấn tối ưu

- Khi duyệt toàn bảng, query optimizer đọc tất cả các hàng trong bảng, và trích ra các hàng thỏa mãn tiêu chuẩn điều kiện truy vấn.
- Việc duyệt bảng phát ra nhiều thao tác I/O đĩa, tiêu tốn nhiều tài nguyên hơn.
- Phương pháp duyệt bảng (table scan) có thể là phương pháp hiệu quả nhất nếu bảng kết quả của truy vấn chứa hầu hết các hàng có trong bảng.



INDEX và truy vấn tối ưu

- Khi query optimizer sử dụng index, nó dò tìm cột khóa của chỉ mục, tìm vị trí lưu trữ của hàng trong bảng và trích ra hàng dữ liệu thỏa mãn điều kiện truy vấn.
- Việc dò tìm chỉ mục nhanh hơn nhiều so với duyệt bảng, vì chỉ mục chỉ chứa 1 số cột và các hàng trong chỉ mục đã được sắp xếp.



INDEX và truy vấn tối ưu

- Nếu không có chỉ mục, query optimizer phải sử dụng phương pháp duyệt bảng
- SQL Server cung cấp công cụ **Database Engine Tuning Advisor** giúp phân tích môi trường database và chọn index phù hợp.



INDEX và truy vấn tối ưu

- Chỉ mục phức có thể là *clustered* hay *non-clustered* mà cột khóa của nó gồm nhiều cột.
- Thực tế chỉ mục phức rất thông dụng.
- Thứ tự các cột trong chỉ mục phức là quan trọng. Để sử dụng chỉ mục phức, điều kiện dò tìm phải bao gồm các cột chỉ mục từ trái sang phải.



INDEX và truy vấn tối ưu

- Ví dụ: nếu có 1 chỉ mục phức mà khóa bao gồm lastname, firstname, việc dò tìm theo firstname sẽ không thể nhanh được nếu dùng chỉ mục, nhưng nếu dò tìm theo lastname, hay lastname và firstname thì chỉ mục sẽ được sử dụng rất hiệu quả.



INDEX và truy vấn tối ưu

- Chỉ mục unique clustered được tạo tự động khi các ràng buộc PRIMARY KEY và UNIQUE được tạo. Tuy nhiên vẫn có thể tạo chỉ mục unique là non-clustered.
- Thực tế ràng buộc unique và chỉ mục unique chỉ là một, chỉ cần tạo 1 trong loại.
- Sự khác nhau cơ bản giữa unique constraint/index và primary key là primary key không cho phép giá trị null, còn unique constraint/index cho phép 1 giá trị null.



Chỉ mục INDEX

- Thiết kế chỉ mục không tốt, hoặc không dùng chỉ mục đều là nguyên nhân cơ bản cho việc “bottlenecks” cho các ứng dụng của database.
- Chọn lựa đúng chỉ mục cần phải xét sự cân đối giữa tốc độ truy vấn và chi phí cập nhật.
- Các chỉ mục ít cột đòi hỏi không gian đĩa ít và chi phí bảo trì thấp.
- Các chỉ mục nhiều cột có thể hỗ trợ cho nhiều truy vấn hơn .



Chỉ mục INDEX

- Có thể mở rộng chức năng của chỉ mục nonclustered bằng cách thêm cột không khóa vào mức lá của chỉ mục nonclustered . Nhờ đó chỉ mục sẽ hỗ trợ (cover) nhiều truy vấn hơn.
- Các lợi điểm khi bao hàm cột không khóa vào chỉ mục:
 - Cột không khóa có kiểu dữ liệu không được dùng trong cột khóa của chỉ mục.
 - Database Engine không tính các cột không khóa được bao hàm vào chỉ mục khi đếm số cột khóa của chỉ mục.



Chỉ mục INDEX

- Chỉ mục nonclustered bị hạn chế về kích cỡ:
 - Số cột khóa tối đa là 16
 - Kích cỡ khóa chỉ mục tối đa là 900 bytes



Chỉ mục INDEX

- Chỉ mục có cột không khoá included có thể cải thiện đáng kể tốc độ thực thi truy vấn khi tất cả cột trong truy vấn đều có mặt trong chỉ mục. Query optimizer có thể định vị tất cả giá trị cột ngay bên trong truy vấn, không cần truy vấn đến bảng dữ liệu nữa → số thao tác I/O giảm.



Chỉ mục INDEX

- Giả sử muốn tạo chỉ mục cho 3 cột sau trong bảng Document của DB AdventureWorks
 - Title nvarchar(50)
 - Revision nchar(5)
 - FileName nvarchar(400)
- Chỉ mục chứa 3 cột này vượt quá 900 byte. Để khắc phục hạn chế này nên tạo chỉ mục có trường không khoá được INCLUDE vào.

```
CREATE INDEX IX_Document_Title  
ON Production.Document (Title, Revision)  
INCLUDE (FileName);
```



Chỉ mục INDEX

Query in which the column predicate is one of these	Query description and example	Index to consider
Exact match to a specific value	<p>Searches for an exact match in which the query uses the WHERE clause to specify a column entry with a specific value. For example:</p> <pre>SELECT EmployeeID, Title FROM HumanResources.Employee WHERE EmployeeID = 228;</pre>	Nonclustered or clustered index on the EmployeeID column.
Exact match to a value in an IN (x,y,z) list	<p>Searches for an exact match to a value in a specified list of values. For example:</p> <pre>SELECT EmployeeID, Title FROM HumanResources.Employee WHERE EmployeeID IN (288, 30, 15);</pre>	Nonclustered or clustered index on the EmployeeID column.

Chỉ mục INDEX

Query in which the column predicate is one of these	Query description and example	Index to consider
Range of values	<p>Searches for a range of values in which the query specifies any entry that has a value between two values. For example:</p> <pre>SELECT ProductModelID, Name FROM Production.ProductModel WHERE ProductModelID BETWEEN 1 and 5;</pre> <p>Or</p> <pre>WHERE ProductModelID >= 1 AND ProductModelID <= 5</pre>	Clustered or nonclustered index on the ProductModelID column.
Join between tables	<p>Searches for rows in a table that match a row in another table based on a join predicate. For example:</p> <pre>SELECT a.ProductAssemblyID, b.Name, a.PerAssemblyQty FROM Production.BillofMaterials AS a JOIN Production.Product AS b ON a.ProductAssemblyID = b.ProductID WHERE b.ProductID = 900;</pre>	Nonclustered or clustered index on the ProductID and ProductAssemblyID columns.

Chỉ mục INDEX

Query in which the column predicate is one of these	Query description and example	Index to consider
LIKE comparison	<p>Searches for matching rows that start with a specific character string such as 'abc%'. For example:</p> <pre>SELECT CountryRegionCode, Name FROM Person.CountryRegion WHERE Name LIKE N'D%'</pre>	Nonclustered or clustered index on the Name column.
Sorted or aggregated	<p>Requires an implicit or explicit sort order or an aggregation (GROUP BY). For example:</p> <pre>SELECT a.WorkOrderID, b.ProductID, a.OrderQty, a.DueDate FROM Production.WorkOrder AS a JOIN Production.WorkOrderRouting AS b ON a.WorkOrderID = b.WorkOrderID ORDER BY a.WorkOrderID;</pre>	<p>Nonclustered or clustered index on the sorted or aggregated column.</p> <p>For sort columns, consider specifying the ASC or DESC order of the column.</p>
PRIMARY KEY or UNIQUE constraint	<p>Searches for duplicates of new index key values in insert and update operations, to enforce PRIMARY KEY and UNIQUE constraints. For example:</p> <pre>INSERT INTO Production.UnitMeasure (UnitMeasureCode, Name, ModifiedDate) VALUES ('OZ1', 'OuncesTest', GetDate());</pre>	Clustered or nonclustered index on the column or columns defined in the constraint.

Chỉ mục INDEX

Query in which the column predicate is one of these	Query description and example	Index to consider
UPDATE or DELETE operation in a PRIMARY KEY/FOREIGN KEY relationship	Searches for rows in an update or delete operation in which the column participates in a PRIMARY KEY/FOREIGN KEY relationship, with or without the CASCADE option.	Nonclustered or clustered index on the foreign key column.
Column is in the select list but not in the predicate.	<p>Contains one or more columns in the select list that are not used for searching and lookups. For example:</p> <pre>SELECT Title, Revision, FileName FROM Production.Document WHERE Title LIKE N'%Maintenance%' AND Revision >= 0';</pre>	Nonclustered index with FileName specified in the INCLUDE clause.



Chỉ mục INDEX

- Giả sử có truy vấn sau:

```
USE AdventureWorks;
```

```
GO
```

```
SELECT AddressLine1, AddressLine2, City,  
       StateProvinceID, PostalCode
```

```
FROM Person.Address
```

```
WHERE PostalCode BETWEEN N'98000' and N'99999';
```

Hãy thiết kế chỉ mục hỗ trợ truy vấn này??



Chỉ mục INDEX

```
CREATE INDEX IX_Address_PostalCode  
ON Person.Address (PostalCode)  
INCLUDE (AddressLine1, AddressLine2, City,  
StateProvinceID);
```