



# INDEXING

Tim Pengajar Manajemen Basis Data – Teknik Informatika ITS

# DATABASE PERFORMANCE

Menurut Anda, apakah yang dimaksud?

Sebagian besar dari kita berpendapat: seberapa **CEPAT** sebuah database menampilkan hasil kueri

Performance dapat di-*tuning* untuk mendapatkan dataset dengan cara yang paling efisien

Salah satu cara utama untuk mempercepat akses ke data adalah dengan menggunakan INDEX

# Pencarian Data Pada Database

(sebelum membahas Index)

- Menggunakan metode **Table-scan** untuk menampilkan hasil query
- Data dicek satu-persatu
- Analogi:

“**Table scan**” dapat diumpamakan seperti mencari sebuah arti kata dalam pada sebuah buku yang tidak memiliki indeks huruf.

- Table-scan akan mencari data pada tabel database dari awal hingga menemukan data yang dicari.

# Analogi: Pencarian kata pada sebuah buku

## Buku dengan index

- Mencari arti kata 'mobil' di Kamus Besar Bahasa Indonesia (KBBI) :
  - membuka kamus berdasarkan index hurufnya
  - mencari huruf 'm'
  - meneruskan pencarian hingga menemukan halaman yang mengandung kata 'mo'
  - mengakhiri proses pencarian hingga didapat kata "mobil"

## Buku Tanpa Index

- mencari dari awal halaman hingga menemukan kata yang dicari.

# Index

- Index merupakan pointer yang menunjuk pada data di sebuah tabel
- Analogi:
  - Coba perhatikan indeks pada sebuah buku:
    - Buku dicetak sesuai dengan urutan halaman
    - Indeks buku ditulis dengan urutan kata kunci atau subjek tertentu
  - Indeks buku lebih cepat menemukan data (lebih efisien)
    - Ketika menambah atau mengubah isi buku, maka indeks semula tidak valid, harus diupdate

# Contoh Pencarian Data

Tabel Karyawan

ID	Nm_Depan	Nm_Belakang
101	Budi	Sutanto
102	Agus	Budiman
103	Fira	Andriani
104	Daniel	Evan
105	Agus	Prasetyo
106	Mia	Sirena
107	Agus	Budi
108	Nita	Panjaitan
109	Agus	Pambudi

Mencari karyawan dengan  
Nama depan Agus

# Pencarian Data: Table-scan (tanpa index)

Tabel Karyawan

ID	Nm_Depan	Nm_Belakang
101	Budi	Sutanto
<b>102</b>	<b>Agus</b>	<b>Budiman</b>
103	Fira	Andriani
104	Daniel	Evan
<b>105</b>	<b>Agus</b>	<b>Prasetyo</b>
106	Mia	Sirena
<b>107</b>	<b>Agus</b>	<b>Budi</b>
108	Nita	Panjaitan
<b>109</b>	<b>Agus</b>	<b>Pambudi</b>

Mencari nama Agus mulai dari ID 101 sd ID 109, semua baris dicek satu-persatu

# Pencarian Data: dengan Index

Tabel Karyawan

ID	Nm_Depan	Nm_Belakang
101	Budi	Sutanto
102	Agus	Budiman
103	Fira	Andriani
104	Daniel	Evan
105	Agus	Prasetyo
106	Mia	Sirena
107	Agus	Budi
108	Nita	Panjaitan
109	Agus	Pambudi

Menambahkan index pada kolom Nm\_Depan, sehingga urut sesuai Nama Depan Karyawan, pencarian lebih cepat

ID	Nm_Depan	Nm_Belakang
<b>102</b>	<b>Agus</b>	<b>Budiman</b>
<b>105</b>	<b>Agus</b>	<b>Prasetyo</b>
<b>107</b>	<b>Agus</b>	<b>Budi</b>
<b>109</b>	<b>Agus</b>	<b>Pambudi</b>
101	Budi	Sutanto
104	Daniel	Evan
103	Fira	Andriani
106	Mia	Sirena
108	Nita	Panjaitan



# Bagaimana Data dimasukkan ke c?

- Data yang dimasukkan ke database tidak terurut
- Data disimpan pada sebuah page (1 page = 8 Kb data)
- Page disimpan pada sebuah extend (1 extend= 8 pages)

# Ilustrasi Pencarian Data

- pencarian data akan dimulai dari pages pertama hingga pages terakhir dari sebuah extent, jika tidak ditemukan maka akan dilanjutkan ke extent berikutnya.
- Kelemahan : saat memasukkan data pada tabel tertentu, data tidak secara otomatis disimpan secara terurut, namun disimpan pada pages yang masih bisa menyimpan data.

- Jadi saat memasukkan data yang harusnya berada di urutan 2 pada tabel, data ini akan disimpan pada pages terakhir yang masih bisa menyimpan data.
- Hal inilah yang seringkali membuat proses pencarian data menjadi lebih lama pada database yang tidak memiliki index, terutama pada database dengan skala OLDB (Online Large Database) dan VLDB (Very Large Database).

# Solusi: Index

- Perlu diingat, setelah Index diterapkan, data tidak akan terurut secara fisik.
- Hal ini berarti, data tidak diurutkan secara terurut pada harddisk.
- Data pada tabel database Anda akan terurut secara **logical** pada level pages dan extent.



# PENGERTIAN INDEX

# Index

- *Index* merupakan struktur data tersendiri yang tidak bergantung kepada struktur tabel.
- Setiap *index* terdiri dari nilai kolom dan penunjuk (atau ROWID) ke baris yang berisi nilai tersebut.
- Penunjuk tersebut secara langsung menunjuk ke baris yang tepat pada tabel, sehingga menghindari terjadinya *full table-scan*.

# Kapan Index digunakan?

- Kolom sering digunakan dalam klausa *WHERE* atau dalam kondisi *join*
- Kolom berisi nilai dengan jangkauan yang luas
- Kolom berisi banyak nilai *null*
- Beberapa kolom sering digunakan dalam klausa *WHERE* atau dalam kondisi *join*
- Tabel berukuran besar dan sebagian besar *query* menampilkan data kurang dari 2-4%.

# Index tidak diperlukan jika:

- Table berukuran kecil
- Kolom tidak sering digunakan sebagai kondisi dalam *query*
- Kebanyakan query menampilkan data lebih dari 2-4% dari seluruh data
- Table sering di-*update*





# JENIS INDEX

# Index

- *clustered index*
- *non-clustered index*

# Clustered Index

- *Clustered index dapat diumpamakan seperti index huruf pada sebuah kamus.*
- Saat sebuah data baru dimasukkan, maka database akan memaksa untuk memasukkan data tersebut pada urutan yang seharusnya.
- Agar data tersimpan sesuai urutan yang kita inginkan, kita dapat menggunakan clustered index pada kolom tabel yang paling sering diakses oleh user
  - Catatan: pembuatan kolom tabel harus spesifik.

ID	NamaLengkap	TempatTanggal Lahir	Alamat
Int	Varchar(50)	Varchar(50)	Varchar(100)
105..	Adi Sasono	Bandung, 28 Oktober 1980	Jl. Sangkuriang 9 Bandung 40116
106..	Budi Sucipto	Sukabumi, 9 Januari 1981	Jl. Teuku Umar 48 Bandung 40117
107..			
108..			

**Tabel 1.** TabelKTP seperti ini kurang spesifik dan dapat menyebabkan masalah saat dilakukan pencarian data

ID	NamaDepan	NamaBelakang	TempatLahir	TanggalLahir	Alamat	Kota	KodePos
Int	Varchar(20)	Varchar(20)	Varchar(15)	Smalldatetime	Varchar(50)	Varchar(15)	Int
105..	Adi	Sasono	Bandung	28-10-1980	Jl. Dago 4	Bandung	40116
106..	Budi	Sucipto	Sukabumi	9-1-1981	Jl. Otista 48	Bandung	40117
107..							
108..							

**Tabel 2.** TabelKTP yang kolom-kolomnya telah dibuat lebih spesifik

# Clustered Index

- Clustered index hanya bisa diterapkan sebanyak 1 kali pada 1 tabel, dan secara otomatis, sebuah primary key juga akan menjadi clustered index pada tabel tersebut.
- Clustered index sebaiknya diterapkan pada kolom tabel yang paling sering digunakan pada saat pencarian data.

# Sintaks Clustered Index

```
CREATE [ UNIQUE ] [ CLUSTERED ] INDEX index_name
ON <object> ( column [ ASC | DESC ] [ ,...n ] )
[ INCLUDE ( column_name [ ,...n ] ) ]
[ WITH ( <relational_index_option> [ ,...n ] ) ]
[ ON { partition_scheme_name ( column_name )
| filegroup_name
| default
}
][ ; ]
```

# Contoh:

- CREATE CLUSTERED INDEX  
ci\_namabelakang ON  
TabelKTP>NamaBelakang);

# Non-clustered Index

- non-clustered index dapat diumpamakan seperti sebuah daftar indeks pada buku.
- Setelah menemukan kata dari index, langkah berikutnya adalah mencari kata pada kamus, sesuai dengan petunjuk dari index
- Ilustrasi tersebut sesuai dengan non-clustered index
- Non-clustered index **berisi pointer-pointer** yang menunjukkan lokasi sesungguhnya dari data

## INDEX

### Symbols

\* (asterisk) key, 23  
/ (forward slash) key, 23  
– (minus) key  
    deleting node with, 233  
    num box calculations with, 23  
+ (plus) key  
    adding node with, 233  
    duplicating selected objects with, 52  
    num box calculations with, 23  
3D effects. *See* Interactive Extrude Tool

### A

acceleration  
    Blend effects and, 561–562  
    child object markers for, 566  
    Contour effects and, 585–587  
Add Language Code dialog, 413  
Add merge mode, 616  
adding  
    extrude lighting, 682–683  
    guidelines, 139  
    nodes, 233  
    tabs, 370

Advanced Separation Settings dialog, 767  
aligning  
    nodes, 236  
    paragraphs, 368  
aligning and distributing objects, 289–296  
    about Align And Distribute dialog, 289–290  
    Distribute tab options for, 294–296  
    hot keys for aligning objects, 294  
    options on Align tab, 290–294  
alpha channels, 734, 735, 740–741  
ALT comments, 801  
altitude for Bevel effects, 629  
amplitude  
    about Distortion effects, 542  
    Zipper distortion, 543–545  
analogous colors, 439  
anchored dockers, 28  
anchoring drop shadows to objects, 634  
AND, OR, and XOR merge mode, 618  
angle option for separations, 765–766

application window  
    appearance of drawing window in, 20  
    features of, 18–20  
applications  
    CorelDRAW's compatibility with other, 8–10  
    font, 325  
arcs  
    drawing lines as, 228–229  
    shaping ellipses like, 199, 200–201  
arrowheads  
    applying, 479–480  
    customizing, 480  
    drawing and saving, 481–483  
    options for, 483  
    reversing direction of, 235  
artifacting and anti-aliasing, 699–700  
Artistic Media Tool, 220–226  
    about, 220  
    applying to special fonts, 393–397  
    breaking paths for, 393, 398  
    Brush mode for, 222–223  
    Calligraphy or Pressure mode



# Non-clustered Index

- *non-clustered index dapat diimplementasikan sebanyak 249 buah pada sebuah tabel.*

- Sintaks:

```
CREATE [ UNIQUE ] [ NONCLUSTERED ] INDEX index_name
ON <object> ( column [ ASC | DESC ] [ ,...n ] )
[ INCLUDE ( column_name [ ,...n ] ) ]
[ WITH ( <relational_index_option> [ ,...n ] ) ]
[ ON { partition_scheme_name ( column_name )
| filegroup_name
| default
}
][ ; ]
```

# Non-clustered Index

- Misalnya, jika pada tabel KTP pada database DataKaryawan, parameter yang juga sering digunakan dalam pencarian data (selain nama belakang) adalah tanggal lahir, maka dapat diimplementasikan *non-clustered index* dengan cara sebagai berikut :
- `CREATE NONCLUSTERED INDEX nci_tanggallahir ON TabelKTP(TanggalLahir);`

# Jenis Index yang didukung Oracle:

Index	Penggunaan
B* - tree index	CREATE INDEX nama_pegawai_idx ON pegawai(nama);
Bitmap index	CREATE BITMAP INDEX jenis_kelamin_idx ON biodata (jenis_kelamin);
Function-based index	CREATE INDEX total_gaji_idx ON penggajian (gaji_pokok + bonus);
Partitioned index	
Revers Key Index	CREATE INDEX nomor_ktp_idx ON biodata (nomor_ktp) REVERSE;
Text index	

# Tugas:

[kelompok masing-masing tiga orang]

- Buat resume mengenai masing-masing jenis Index yang didukung pada Oracle
- Kemudian buat contoh implementasi Index pada studi kasus ETS