**LAPORAN PRAKTIKUM**

**PRAKTIKUM 5:**

**“RESPONSI”**



**Disusun Oleh :**

Oktaviana Sadama Nur Azizah

24060121130060

PRAKTIKUM MANAJEMEN BASIS DATA

LAB A2

**DEPARTEMEN ILMU KOMPUTER / INFORMATIKA**

**FAKULTAS SAINS DAN MATEMATIKA**

**UNIVERSITAS DIPONEGORO**

**SEMARANG**

**2023**

# BAB I PENDAHULUAN

1. **Rumusan Masalah** 
   1. Di dalam database RESPONSI terdapat dua buah tabel, yaitu:
      * + 1. PRODUK (NoProduk, NamaProduk, Kategori, Warna, Stok, StokMin, HargaPokok, HargaJual).
          2. JUAL (NoFaktur, NoProduk, TglFaktur, Jumlah).
   2. Buat trigger untuk:
      * + 1. Mengupdate nilai harga jual menjadi sebesar harga pokok + 5% \* harga pokok.
          2. Mengupdate nilai stok pada tabel produk ketika produk tersebut berhasil dijual dan sekaligus menampilkan pesan ketika mengalami kondisi stok dibawah stok minimum.
   3. Buat Store Procedure untuk:
      * + 1. Menampilkan periode waktu (bulan/ tahun), total pendapatan, dan total laba per bulan.
          2. Menampilkan data produk terjual (NoProduk, NamaProduk, JumlahTerjual) per masing-masing produk berdasarkan inputan periode waktu tertentu.
   4. Berdasarkan database di atas, kebutuhan sistem yang sedang dikembangkan antara lain meliputi:

Pada kedua tabel di atas sering dikenai perintah SQL seperti:

SELECT Produk.NoProduk, NamaProduk

FROM Produk, Jual

WHERE Produk.NoProduk = Jual.NoProduk

AND Kategori = ‘Alat Rumah Tangga’ ……………………...………… (query 1)

Selain hal di atas pada tabel-tabel tersebut juga sering dikenakan perintah SQL untuk menampilkan data produk yang terjual untuk suatu periode waktu tertentu.

* + - * 1. Tuliskan indeks-indeks yang telah dibuat dalam database RESPONSI! Jelaskan!
        2. Tentukan indeks-indeks tambahan yang diperlukan, field yang digunakan sebagai search key, tipe indeks, dan jelaskan untuk masing-masing indeks tersebut!
        3. Bagaimana query optimizer mengeksekusi perintah SQL yang disebutkan di atas (query 1), setelah ditambahkan indeks? Jelaskan!
        4. Implementasikan indeks yang sudah anda desain pada soal b!
  1. Gunakan fasilitas yang ada pada SQL Server untuk menyimpan:
     + - 1. Data trigger dan indeks yang dilampirkan pada tabel dengan nama ***“trigger\_index.sql”***
         2. Data stored procedure yang dibuat pada database anda dengan nama ***“stor\_proc.sql”***

1. **Tujuan** 
   1. Mahasiswa mampu mengimplementasikan trigger di dalam database.
   2. Mahasiswa mampu mengimplementasikan store procedure di dalam database.
   3. Mahasiswa mampu mengimplementasikan indeksing di dalam database.

# BAB II DASAR TEORI

1. **Trigger**

Dalam Database Management System (DBMS), trigger merupakan kumpulan script yang berhubungan dengan table, view, ataupun schema yang dijalankan secara otomatis ketika terdapat event yang dijalankan. Event tersebut meliputi operasi yang biasa dilakukan dalam mengolah database, seperti:

1. Data Manipulation Language (DML) yang meliputi DELETE, INSERT, atau UPDATE.
2. Data Definition Language (DDL) yang meliputi CREATE, ALTER, atau DROP.
3. Operasi database lainnya, seperti SERVERERROR, LOGON, LOGOFF, STARTUP, atau SHUTDOWN.

Secara sederhana, perintah untuk membuat trigger adalah sebagai berikut:

|  |
| --- |
| **CREATE TRIGGER** nama\_trigger  **ON** nama\_table  [ BEFORE | AFTER] [ INSERT | UPDATE | DELETE ]  AS  BEGIN  // trigger body  END; |

Keterangan:

* 1. Nama\_trigger

Nama trigger yang dibuat sesuai dengan karakteristik penamaan.

* 1. Nama\_table

Menunjukkan table yang akan dilakukan trigger didalamnya.

* 1. [ BEFORE | AFTER ]

Menunjukkan waktu untuk mengeksekusi trigger secara otomatis, apakah sebelum atau sesudah perubahan pada row data table. Sehingga pilihannya adalah AFTER atau BEFORE.

* 1. [ INSERT | UPDATE | DELETE ]

Digunakan untuk menentukan event yang menyebabkan terjadinya trigger, pilihan event tersebut terdiri dari INSERT, UPDATE, dan DELETE.

* 1. Trigger\_body

Menunjukkan statement perintah yang akan otomatis dijalankan jika event sedang aktif.

1. **Store Procedure**

Stored Procedure adalah sebuah fungsi yang berisi kode SQL yang dapat digunakan kembali dengan cara memanggil atau *execute* Stored Procedure yang telah dibuat. Dalam

Stored Procedure juga dapat dimasukkan parameter sehingga fungsi dapat digunakan lebih dinamis berdasarkan parameter tersebut.

Stored Procedure Syntax

|  |
| --- |
| **CREATE PROCEDURE** nama\_prosedur  **AS**  **BEGIN**  SQL\_statement  **END;** |

Pemanggilan Stored Procedure

|  |
| --- |
| **EXEC** nama\_prosedur; |

Adapun kelebihan menggunakan Stored Procedure adalah sebagai berikut:

* + - 1. Berbagi logik dengan aplikasi lainnya

Stored Procedure merangkum fungsionalitas untuk memastikan akses data dan manipulasi koheren antara aplikasi yang berbeda.

* + - 1. Keamanan

Stored Procedure mengisolasi pengguna dari tabel data. Fitur ini memberi kemampuan untuk memberikan akses ke Stored Procedure yang memanipulasi data namun tidak secara langsung ke tabel.

* + - 1. Performa

Stored Procedure dikompilasi yang selanjutnya dicache dan akan digunakan kembali. Dengan begitu waktu respon dan performa menjadi lebih cepat karena Stored Procedure yang sama dieksekusi kembali.

Selain kelebihan Stored Procedure yang sudah disebutkan diatas, terdapat pula kekurangan dari Stored Procedure sebagai berikut:

Peningkatan beban pada database server. Sebagian besar pekerjaan dilakukan di sisi server dan kurang di sisi client.

Pengulangan logika aplikasi di dua tempat yang berbeda: kode bahasa pemrograman dan kode Stored Procedure membuat pemeliharaan aplikasi menjadi lebih sulit.

Migrasi ke sistem manajemen basis data yang berbeda berpotensi menjadi lebih sulit.

Tidak membutuhkan ruang disk tambahan.

1. **Indeks**

Indeks adalah kunci yang dibuat dari satu atau beberapa kolom dalam database yang berguna untuk mempercepat pengambilan baris dalam tabel atau tampilan. Kunci ini membantu database untuk menemukan baris yang terkait dengan nilai kunci dengan cepat.

Adapun dua tipe indeks dalam database adalah sebagai berikut.

* + - 1. Clustered Indeks

Clustered indeks atau yang biasa disebut indeks berkelompok atau berkerumun adalah jenis indeks yang mengurutkan baris data dalam tabel berdasarkan nilai kuncinya. Di database, hanya diperbolehkan terdapat satu clustered indeks per tabel. Adapun ciri dari clustered indeks sebagai berikut.

* Menentukan urutan penyimpanan baris dalam tabel secara keseluruhan.
* Hanya diperbolehkan ada satu clustered indeks dalam satu tabel.
* Akses data lebih cepat.
* Tidak membutuhkan ruang disk tambahan.
  + - 1. Non Clustered Indeks

Non clustered indeks menyimpan data di satu lokasi dan indeks di lokasi lain. Indeks ini berisi pointer ke lokasi data tersebut. Satu tabel dapat memiliki banyak non clustered indeks karena indeks disimpan di tempat yang berbeda. Adapun ciri dari non clustered indeks sebagai berikut.

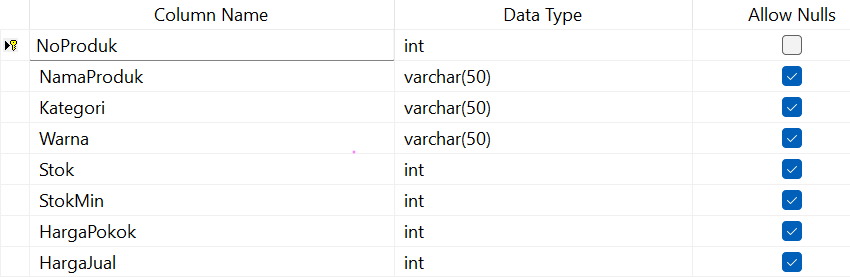
* Menentukan urutan penyimpanan baris dalam tabel dengan bantuan struktur fisik yang terpisah.
* Diperbolehkan terdapat lebih dari satu non clustered indeks dalam satu tabel.
* Akses data lebih lambat dibandingkan dengan clustered indeks.
* Memerlukan ruang disk tambahan untuk menyimpan indeks secara terpisah.

Berikutadalahsyntax yang digunakan untuk mengimplementasikan indeks dalam SQL Server.

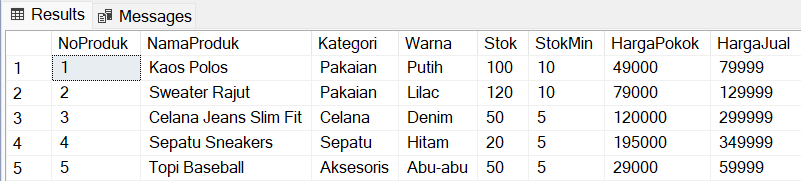
|  |
| --- |
| **USE** database\_name  **CREATE** index\_type INDEX index\_name  **ON** table\_name (column\_name) |

# BAB III PEMBAHASAN

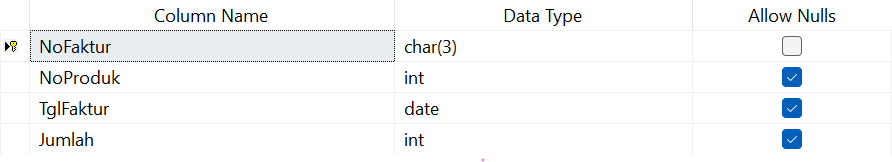
* 1. Di dalam database RESPONSI terdapat dua buah tabel, yaitu:
     + - 1. PRODUK (NoProduk, NamaProduk, Kategori, Warna, Stok, StokMin, HargaPokok, HargaJual).



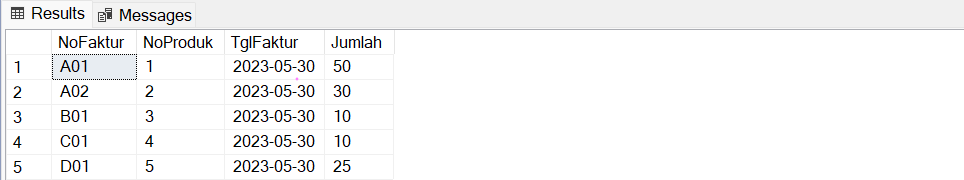
Tabel PRODUK dibuat dimana kolom NoProduk diberi tipe data INT dan di set sebagai Primary Key, kolom NamaProduk diberi tipe data VARCHAR(50), kolom Kategori diberi tipe data VARCHAR(50), kolom Warna diberi tipe data VARCHAR(50), kolom Stok diberi tipe data INT, kolom StokMin diberi tipe data INT, tipe HargaPokok diberi tipe data INT, sedangkan kolom HargaJual diberi tipe data INT. Berikut tabel PRODUK yang sudah isi data.



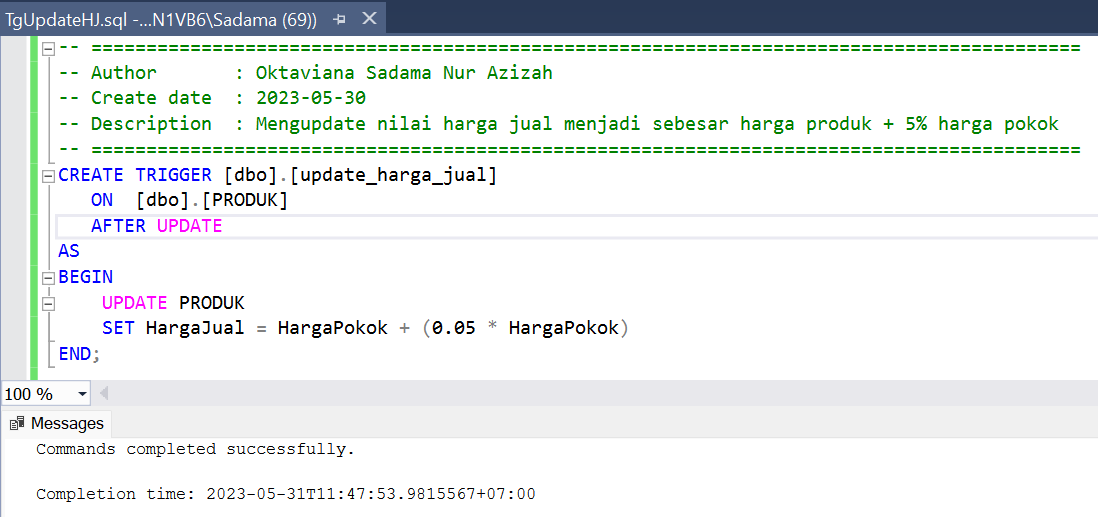
* + - * 1. JUAL (NoFaktur, NoProduk, TglFaktur, Jumlah).



Tabel JUAL dibuat dimana kolom NoFaktor diberi tipe data CHAR(3) dan di set sebagai Primary Key, kolom NoProduk diberi tipe data INT, kolom TglFaktur diberi tipe data DATE, sedangkan kolom Jumlah diberi tipe data INT. Berikut tabel JUAL yang sudah isi data.

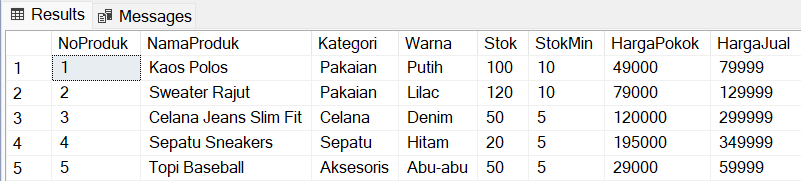


* 1. Buat trigger untuk:
     + - 1. Mengupdate nilai harga jual menjadi sebesar harga pokok + 5% \* harga pokok.



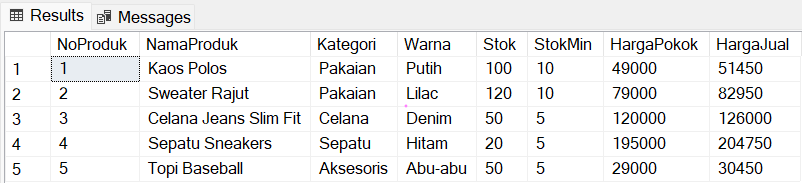
Trigger dibuat dengan perintah CREATE TRIGGER dengan nama trigger ‘update\_harga\_jual’ dan menggunakan perintah AFTER UPDATE. Untuk mengupdate nilai kolom HargaJual di tabel produk sebesar harga pokok + 5% \* harga pokok, digunakan perintah UPDATE PRODUK dan SET HargaJual = HargaPokok + (0.05 \* HargaPokok). Berikut adalah tabel produk sebelum dan sesudah dilakukan update harga jual.

Sebelum Trigger



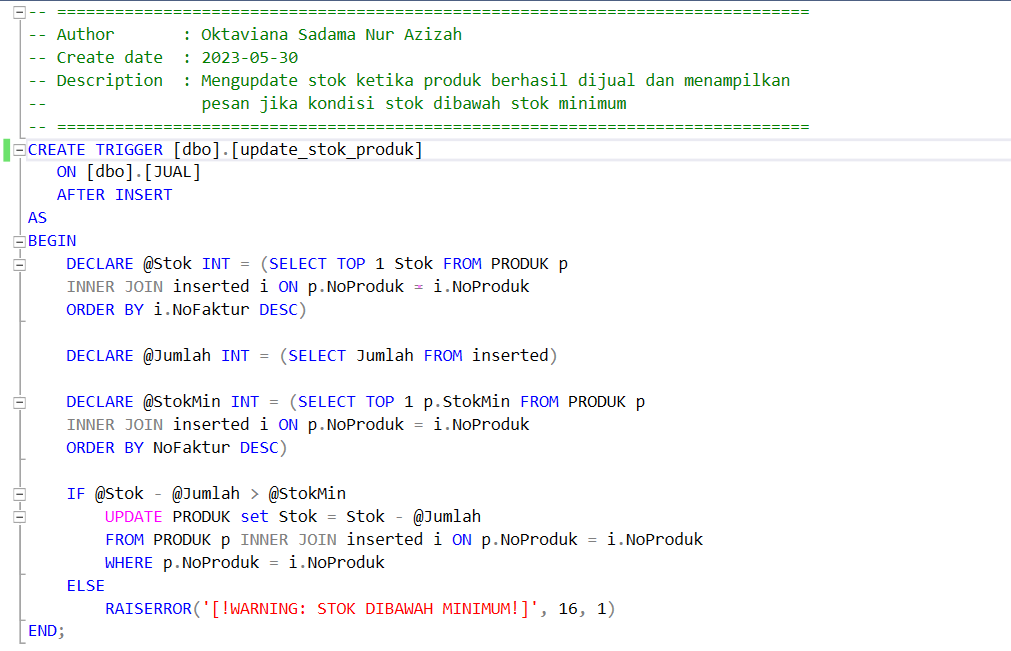


Setelah Trigger





* + - * 1. Mengupdate nilai stok pada tabel produk ketika produk tersebut berhasil dijual dan sekaligus menampilkan pesan ketika mengalami kondisi stok dibawah stok minimum.

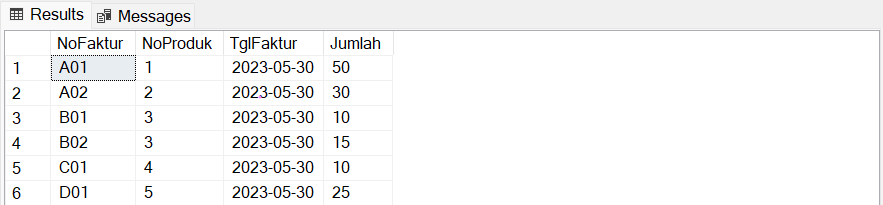


Trigger dibuat dengan perintah CREATE TRIGGER dengan nama trigger ‘update\_stok\_produk’ dan menggunakan perintah AFTER INSERT. Variabel @Stok digunakan untuk menyimpan nilai stok terbaru dari produk yang terlibat dalam operasi INSERT terakhir di tabel JUAL. Variabel ini mendapatkan nilai stok dengan mengambil stok terakhir dari tabel PRODUK yang terhubung dengan baris yang baru di insert menggunakan pernyataan JOIN.

Variabel @Jumlah menyimpan nilai jumlah produk yang diinsert ke dalam tabel JUAL. Variabel ini mendapatkan nilai jumlah dari baris yang baru diinsert menggunakan pernyataan SELECT. Sedangkan variabel @StokMin menyimpan nilai stok minimum terbaru dari produk yang terlibat dalam operasi INSERT terkahir di tabel JUAL. Variabel ini mendapatkan nilai stok minimum terakhir dari tabel PRODUK yang terhubung dengan baris yang baru diinsert menggunakan pernyataan JOIN.

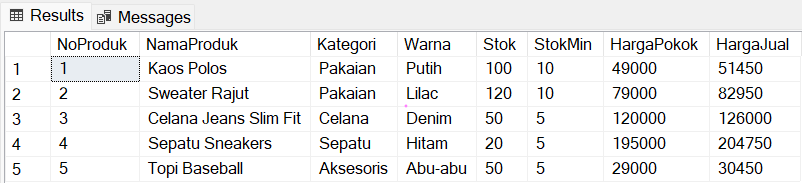
Kemudian digunakan pernyataan kondisional IF @Stok - @Jumlah > @StokMin, apabila terpenuhi maka dilakukan UPDATE jumlah produk. Sedangkan apabila tidak terpenuhi maka akan mengeluarkan teks error [!WARNING: STOK DIBAWAH MINIMUM!].

Inserted Data



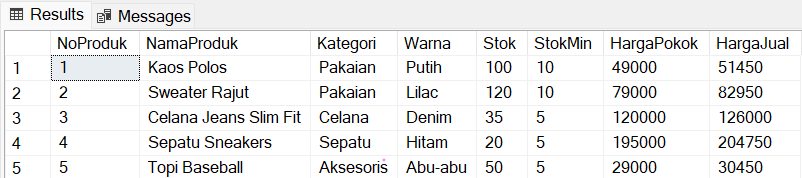


Sebelum Trigger





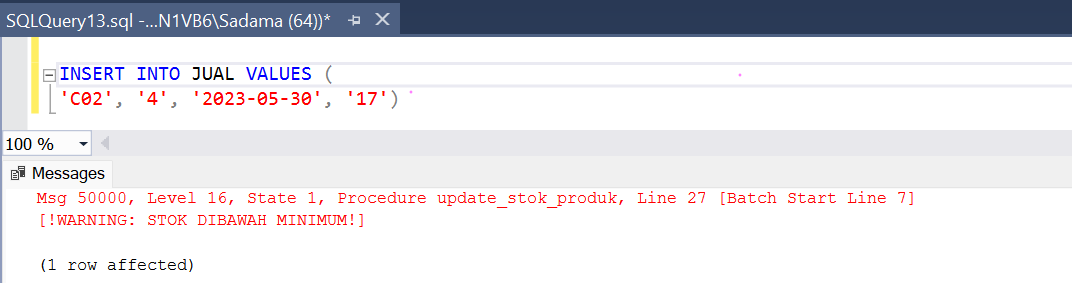
Setelah Trigger



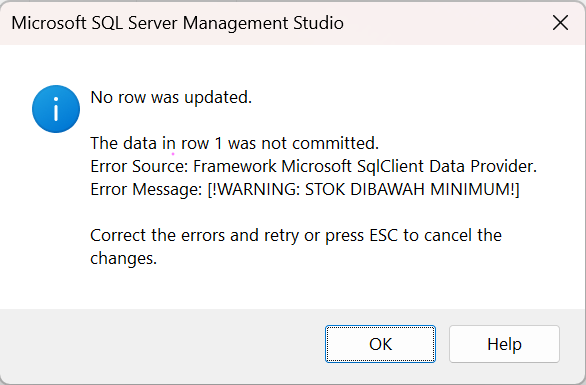


Dilakukan inserting data pada tabel JUAL (‘B02’, 3, ‘2023-05-30’, 15), maka stok barang yang memiliki NoProduk = 3, yaitu Celana Jeans Slim Fit akan mengalami pengurangan jumlah stok menjadi 50 – 15 = 35.

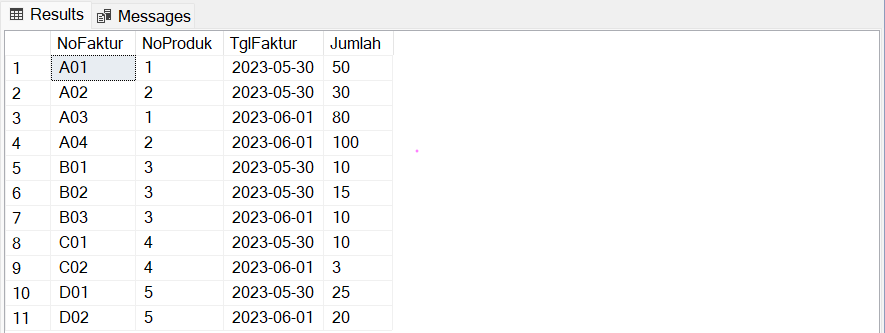
Teks Error



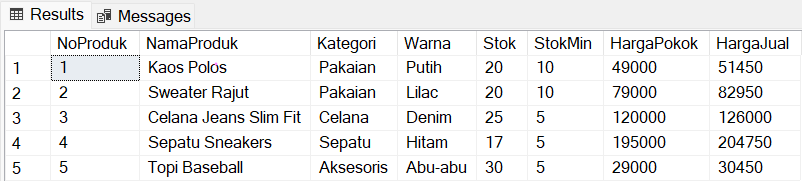
Kemudian dilakukan lagi inserting data pada tabel JUAL seperti gambar diatas. Inserting data dilakukan pada kolom yang memiliki NoProduk = 4, yaitu Sepatu Sneakers yang merefer pada kolom PRODUK. Apabila inserting kolom Jumlah = 17 pada tabel JUAL dilakukan maka sisa stok sepatu menjadi 20 – 17 = 3. Hal ini tidak diperbolehkan karena stok minimum sepatu sneakers yang diperbolehkan adalah 5. Maka error teks akan muncul seperti gambar diatas ataupun seperti gambar dibawah ini.



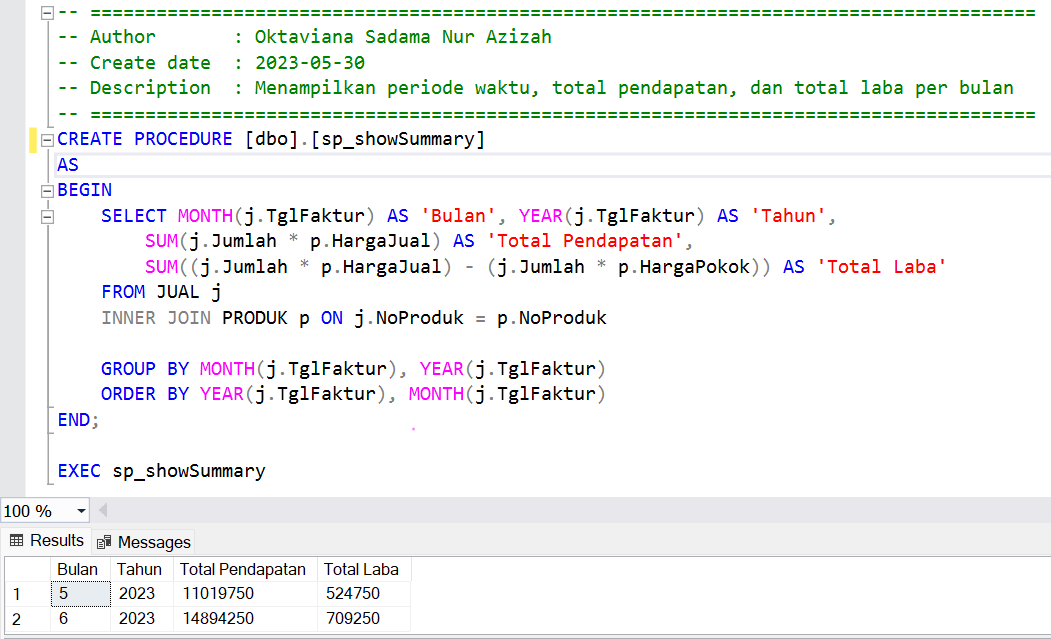
* 1. Buat Store Procedure untuk:
     + - 1. Menampilkan periode waktu (bulan/ tahun), total pendapatan, dan total laba per bulan.







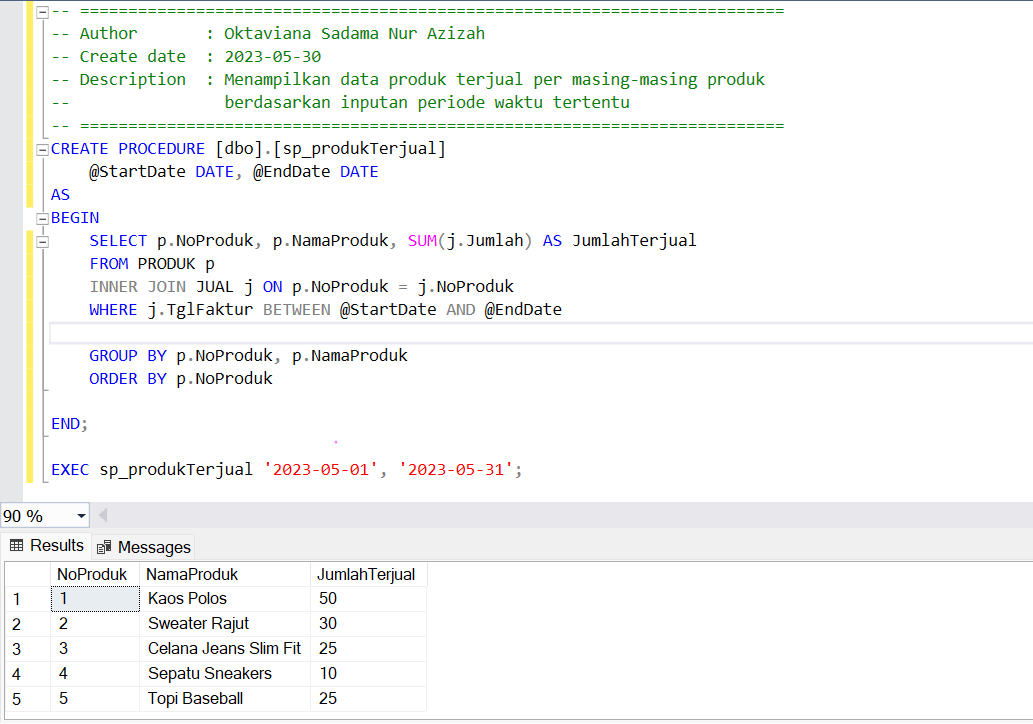
Sebelum membuat Store Procedure, dilakukan inserting data seperti gambar diatas.



Stored Procedure dibuat dengan perintah CREATE PROCEDURE dengan nama ‘sp\_showSummary’. Store Procedure diatas menampilkan kolom periode waktu (bulan dan tahun) dengan perintah MONTH(j.TglFaktur) dan YEAR(j.TglFak-tur), kolom Total Pendapatan dengan rumus SUM(j.Jumlah \* p.Harga Jual), dan kolom Total Laba dengan rumus SUM((j.Jumlah \* p.HargaJual) – (j.Jumlah \* p.HargaPokok)).

Untuk mengubungkan tabel PRODUK dan JUAL maka dilakukan INNER JOIN dengan menghubungkan j.NoProduk = p.NoProduk. Kemudian dilakukan GROUPING dan ORDER BY ASC. Store Procedure dipanggil dengan perintah EXEC sp\_showSummary.

* + - * 1. Menampilkan data produk terjual (NoProduk, NamaProduk, JumlahTerjual) per masing-masing produk berdasarkan inputan periode waktu tertentu.



Stored Procedure dibuat dengan perintah CREATE PROCEDURE dengan nama ‘sp\_produkTerjual’ menggunakan parameter @StartDate dan @EndDate dengan tipe data DATE. Parameter ini digunakan untuk menginputkan periode waktu tertentu. Perintah SELECT digunakan untuk menampilkan NoProduk, NamaProduk dan JumlahTerjual dengan rumus SUM(j.Jumlah).

Perintah INNER JOIN digunakan untuk menghubungkan tabel PRODUK dan JUAL dimana p.NoProduk = j.NoProduk dan dilakukan seleksi dengan perintah WHERE dimana j.TglFaktur BETWEEN @StartDate AND @EndDate. Kemudian dilakukan GROUPING dan ORDER BY ASC. Store Procedure dipanggil dengan perintah EXEC sp\_produkTerjual ‘2023-05-01’, ‘2023-05-31’.

* 1. Berdasarkan database di atas, kebutuhan sistem yang sedang dikembangkan antara lain meliputi:

Pada kedua tabel di atas sering dikenai perintah SQL seperti:

SELECT Produk.NoProduk, NamaProduk

FROM Produk, Jual

WHERE Produk.NoProduk = Jual.NoProduk

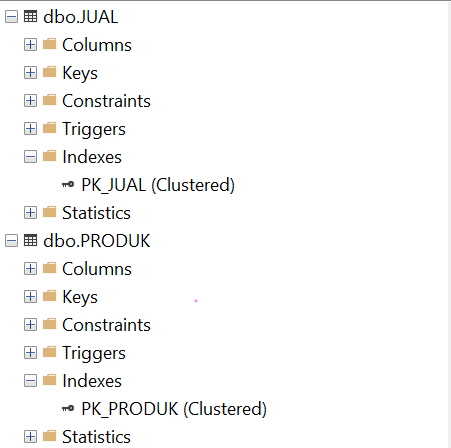
AND Kategori = ‘Alat Rumah Tangga’ ………………………………………… (query 1)

Selain hal di atas pada tabel-tabel tersebut juga sering dikenakan perintah SQL untuk menampilkan data produk yang terjual untuk suatu periode waktu tertentu.

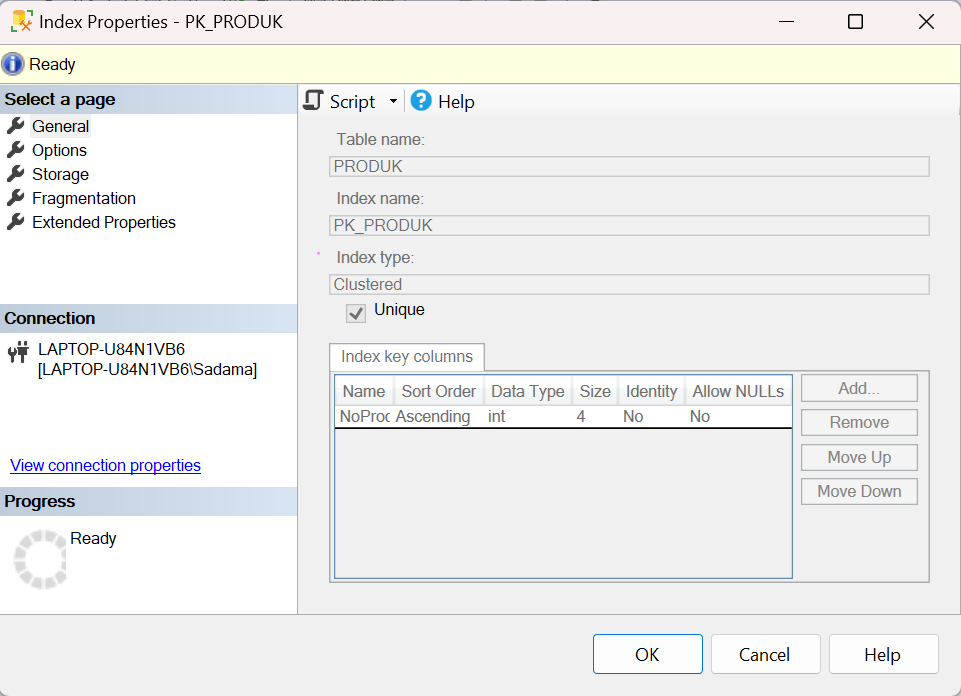
* + - * 1. Tuliskan indeks-indeks yang telah dibuat dalam database RESPONSI! Jelaskan!

Penjelasan:

Clustered Indeks Tabel Produk

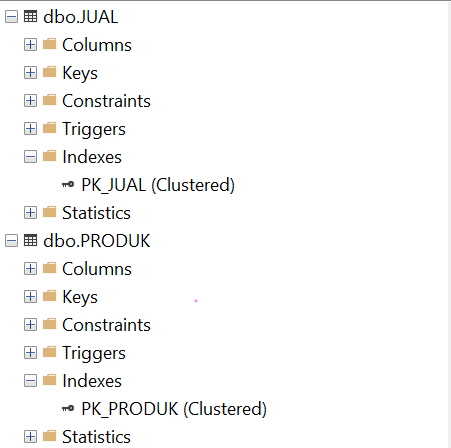




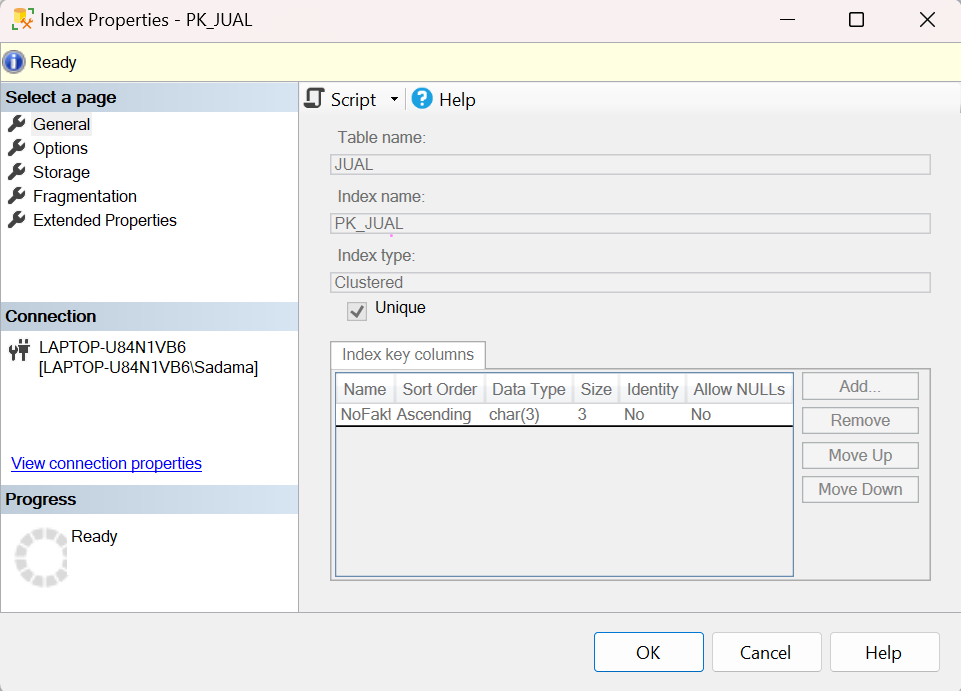


Clustered indeks otomatis terbuat pada tabel PRODUK saat mendifinisikan Primary Key. Field yang digunakan sebagai search key adalah NoProduk.

Clustered Indeks Tabel Jual







Clustered indeks otomatis terbuat pada tabel JUAL saat mendifinisikan Primary Key. Field yang digunakan sebagai search key adalah NoFaktur.

* + - * 1. Tentukan indeks-indeks tambahan yang diperlukan, field yang digunakan sebagai search key, tipe indeks, dan jelaskan untuk masing-masing indeks tersebut!

Penjelasan:

Berdasarkan query 1, indeks tambahan yang diperlukan adalah non clustered indeks sebagai berikut.

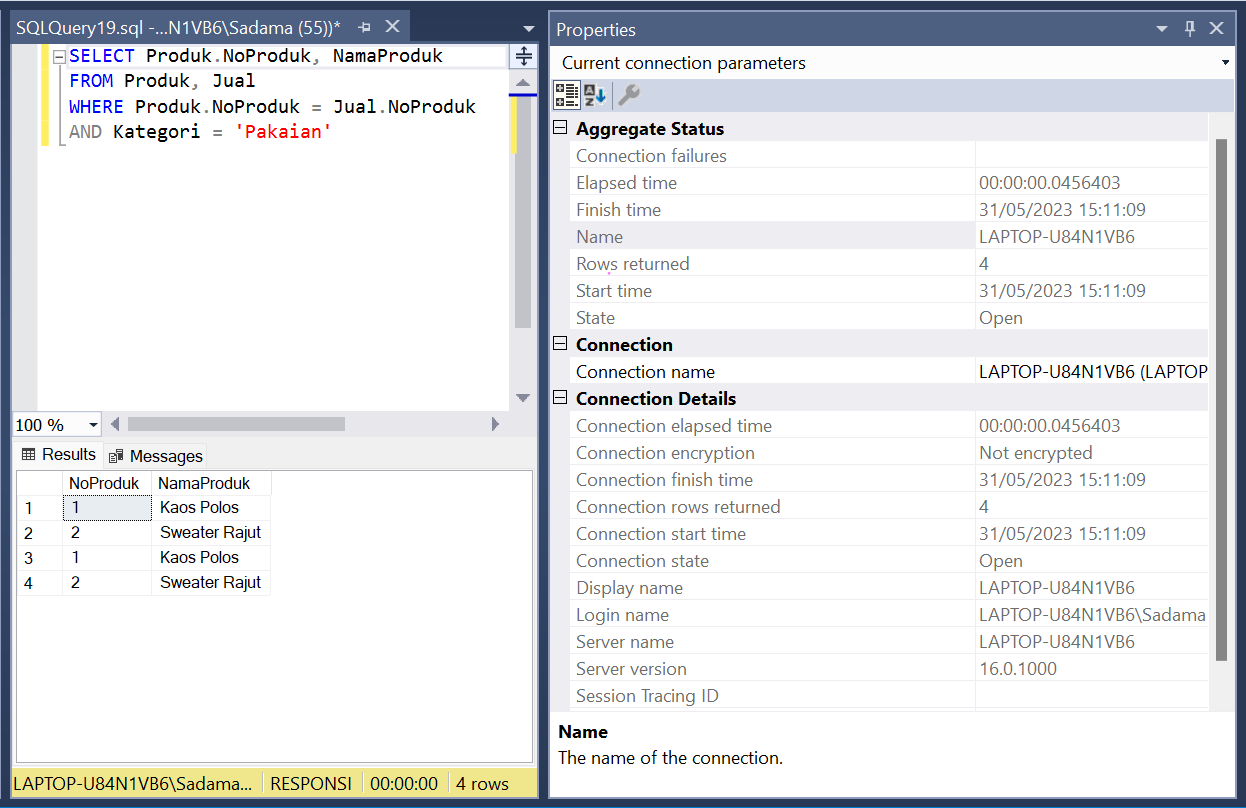
|  |  |
| --- | --- |
| Non Clustered Indeks | |
| Tabel | Field Search Key |
| PRODUK | NoProduk  NamaProduk  Kateogori |
| JUAL | NoProduk |

Non clustered indeks pada tabel PRODUK dan JUAL dibuat untuk mempercepat waktu eksekusi query saat pencarian data.

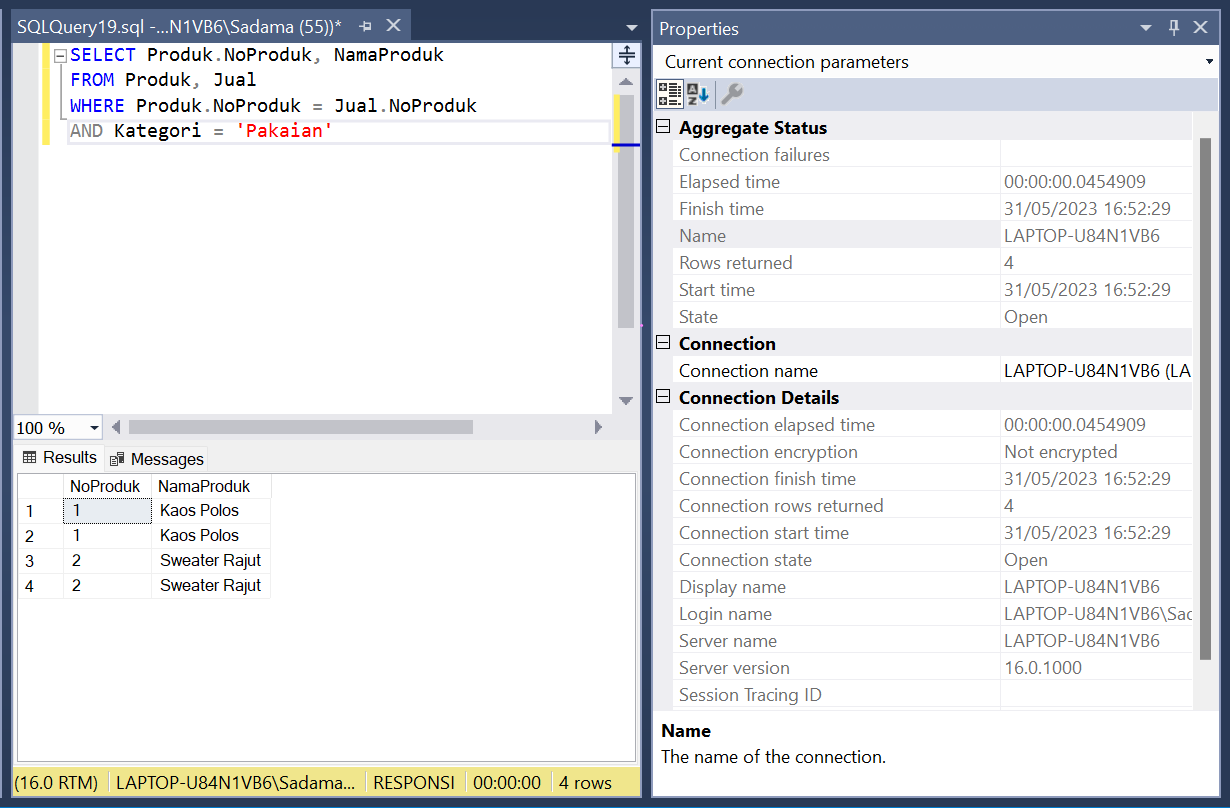
* + - * 1. Bagaimana query optimizer mengeksekusi perintah SQL yang disebutkan di atas (query 1), setelah ditambahkan indeks? Jelaskan!

Penjelasan:

Sebelum ditambahkan non clustered indeks, query 1 membutuhkan elapsed time sebesar 00:00:00.0456403 ditunjukkan seperti gambar berikut.



Setelah ditambahkan non clustered indeks, query 1 hanya membutuhkan elapsed time sebesar 00:00:00.0454909 ditunjukkan seperti gambar berikut.



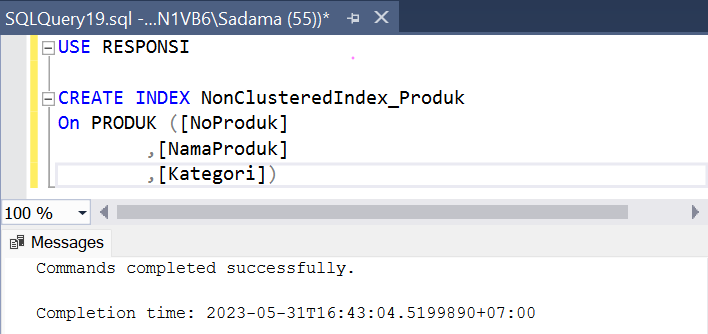
Penambahan non clustered indeks pada kolom NoProduk, NamaProduk, dan Kategori terbukti mengoptimalkan query. Waktu eksekusi yang lebih singkat meningkatkan efektifitas dan kinerja sistem secara keseluruhan.

**Note:** Kategori disesuaikan isi data milik pribadi, disini saya menggunakan Kategori = ‘Pakaian’.

* + - * 1. Implementasikan indeks yang sudah anda desain pada soal b!

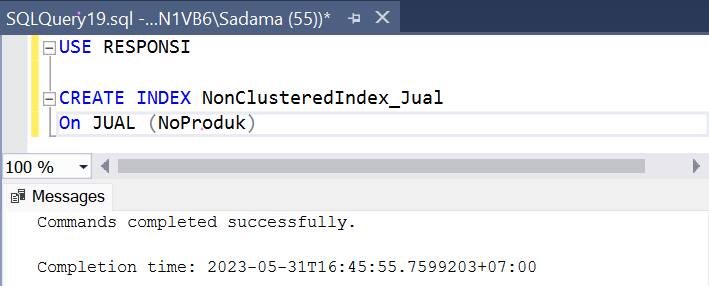
Non Clustered Indeks Prodok

Berikut adalah pembuatan non clustered indeks untuk tabel PRODUK dengan perintah CREATE INDEX [nama\_indeks]. Dalam contoh dibawah ini dibuat non clustered indeks dengan nama NonClusteredIndex\_Produk di kolom NoProduk, NamaProduk, dan Kategori.



Non Clustered Indeks Jual

Berikut adalah pembuatan non clustered indeks untuk tabel JUAL dengan perintah CREATE INDEX [nama\_indeks]. Dalam contoh dibawah ini dibuat non clustered indeks dengan nama NonClusteredIndex\_Jual di kolom NoProduk.



* 1. Gunakan fasilitas yang ada pada SQL Server untuk menyimpan:
     + - 1. Data trigger dan indeks yang dilampirkan pada tabel dengan nama ***“trigger\_index.sql”***
         2. Data stored procedure yang dibuat pada database anda dengan nama ***“stor\_proc.sql”***

**(No.5 terlampir dalam RAR file. Terimakasih 😊)**

# BAB IV

# PENUTUP

# Kesimpulan

# Berdasarkan responsi yang telah dilakukan, didapati beberapa kesimpulan sebagai berikut:

# Trigger merupakan kumpulan script yang berhubungan dengan table, view, ataupun schema yang dijalankan secara otomatis ketika terdapat event yang dijalankan.

# Stored Procedure adalah sebuah fungsi yang berisi kode SQL yang dapat digunakan kembali dengan cara memanggil Stored Procedure yang telah dibuat dengan perintah EXEC. Dalam Stored Procedure juga dapat dimasukkan parameter sehingga fungsi dapat digunakan lebih dinamis berdasarkan parameter tersebut.

# Indeks adalah kunci yang dibuat dari satu atau beberapa kolom dalam database yang berguna untuk mempercepat pengambilan baris dalam tabel. Indeks terdiri dari dua tipe, yaitu clustered indeks dan non clustered indeks.