12/29/2024

Tugas Pertemuan Ke - ∞

tugas tambahan

Nama: rifandy arnas

nim/npm: 232310001

Kelas: TI – 23 – PA (Lab 2)

BASIS DATA

PERTEMUAN ∞

Github: <https://github.com/Rifandy232310001TeknologiInformasi/Basis_Data-Lab>

**Triggering**

1. **Membuat Database dan Tabel:**
   * Tabel utama: orders, orderdetails, products.
   * Tabel audit/log: orderdetails\_audit, order\_log.
2. **Insert Data:**
   * Data dummy dimasukkan untuk tabel orderdetails, orders, dan products.
3. **Trigger untuk Audit dan Validasi:**
   * **Audit Update**: Memasukkan log perubahan ke tabel orderdetails\_audit.
   * **Log Status Pesanan**: Mencatat perubahan status pesanan di order\_log.
   * **Validasi Stok**: Mencegah stok produk menjadi negatif.
4. **Uji Implementasi:**
   * Melakukan update pada data untuk memvalidasi trigger.

**Stored Procedure**

1. **Penambahan Data:**
   * AddNewOrder: Menambah pesanan baru ke tabel orders dan orderdetails.
   * AddNewProduct: Menambah produk baru ke tabel products.
2. **Pembaharuan Data:**
   * UpdateOrderStatus: Mengubah status pesanan dan mencatat perubahan.
   * UpdateProductStock: Memperbarui jumlah stok produk.
3. **Pengambilan Data:**
   * GetOrderDetails: Melihat detail pesanan dan produk terkait.
4. **Contoh Implementasi:**
   * Penggunaan stored procedure disertai contoh sintaks CALL.

**Fungsi (Function)**

1. **Fungsi Dasar:**
   * calculateOrderTotal: Menghitung total harga pesanan berdasarkan orderNumber.
   * isProductAvailable: Mengecek ketersediaan produk.
   * calculateDiscount: Menghitung diskon berdasarkan total pesanan.
   * formatOrderDate: Mengubah format tanggal menjadi DD-MM-YYYY.
2. **Penggunaan dan Implementasi:**
   * Memanggil fungsi untuk menghitung total pesanan, mengecek diskon, dan menampilkan produk tersedia.
   * Format tanggal diterapkan pada tabel orders.
3. **Manajemen Fungsi:**
   * Menampilkan daftar fungsi dengan SHOW FUNCTION STATUS.
   * Menghapus fungsi dengan DROP FUNCTION.

**Transaksi**

1. **Proses Transaksi Baru:**
   * Menambahkan pesanan dan detail pesanan.
   * Melibatkan tabel orders dan orderdetails.
2. **Validasi Stok:**
   * Periksa ketersediaan stok sebelum mengurangi jumlah produk.
   * Gunakan ROLLBACK jika stok tidak mencukupi atau terdapat error.
3. **Audit dan Simulasi Error:**
   * Transaksi mencatat log jika gagal.
   * Simulasi kesalahan dilakukan untuk menguji mekanisme rollback.
4. **Manajemen Prosedur:**
   * Tambahkan pesanan dan catat error jika transaksi gagal menggunakan AddOrder.
   * Daftar prosedur dapat dilihat dengan SHOW PROCEDURE STATUS.

**Pengertian Function**

* Function adalah program tersimpan yang mengembalikan nilai tunggal.
* Berbeda dengan stored procedure, function dapat digunakan langsung dalam pernyataan SQL.

**Cara Membuat Function**

* Gunakan sintaks CREATE FUNCTION, sertakan nama, parameter, tipe data nilai kembali, dan logika di dalam blok BEGIN ... END.
* Gunakan pernyataan RETURN untuk mengembalikan nilai ke pemanggil.

**Best Practices**

* Gunakan nama yang deskriptif.
* Batasi logika yang terlalu kompleks.
* Hindari efek samping seperti INSERT, UPDATE, atau DELETE dalam function.
* Gunakan parameter dengan tipe data yang sesuai.

**Penggunaan Function**

* Function dapat dipanggil dalam pernyataan SQL atau prosedur tersimpan.

**Tujuan Penggabungan Tabel**

* Mengintegrasikan data dari beberapa tabel untuk membuat korelasi antar data.

**Metode Penggabungan:**

* **Join:**
  + **Inner Join:** Menampilkan data yang cocok di kedua tabel.
  + **Left Join:** Menampilkan semua data dari tabel kiri, termasuk yang tidak cocok.
  + **Right Join:** Menampilkan semua data dari tabel kanan, termasuk yang tidak cocok.
  + **Full Join:** Menampilkan semua data dari kedua tabel, baik yang cocok maupun tidak.
  + **Self Join:** Menggabungkan data dari tabel yang sama.
  + **Cross Join:** Menggabungkan semua kombinasi baris dari dua tabel (Cartesian product).
* **Union:**
  + Menggabungkan hasil dari beberapa pernyataan SELECT tanpa duplikasi.
  + **Union All:** Menggabungkan hasil SELECT dengan mencantumkan baris duplikat.
* **Subquery:**
  + Query dalam query lain untuk memfilter, menghitung nilai, atau membuat tabel baru.

**Tugas Praktik SQL:**

* Membuat database toko buku dengan tabel: Pelanggan, Buku, Kategori, Transaksi, dan Detail Transaksi.
* Menampilkan data seperti daftar buku berdasarkan kategori, transaksi di atas nilai tertentu, dan pelanggan yang belum pernah bertransaksi.

**Program Klien SQL**

* Digunakan untuk mengelola basis data (membuat, menghapus, memodifikasi data, menjalankan query).
* Contoh program klien:
  + **SQL Server Management Studio** (Microsoft SQL Server)
  + **MySQL Workbench** (MySQL)
  + **PostgreSQL Command-Line Client**
  + **SQLite3 CLI**
* Pengguna perlu terhubung ke database dan memasukkan perintah SQL melalui shell.

**Fungsi SQL**

* Fungsi digunakan dalam perintah SQL untuk manipulasi data.
* Klasifikasi:
  + **System Function:** Memberikan informasi tentang penggunaan server database.
  + **Aggregate Function:** Melakukan ringkasan data (contoh: AVG, SUM).
  + **Arithmetic Function:** Manipulasi numerik (contoh: +, -, \*, /).
  + **String Function:** Manipulasi teks.
  + **Date Function:** Mengolah data berbasis tanggal.
  + **Logic Function:** Mengevaluasi ekspresi logika (contoh: IF, ISNULL).

**Subquery dalam MySQL**

1. **Definisi**
   * Subquery adalah kueri yang bersarang di dalam kueri lain (SELECT, INSERT, UPDATE, DELETE).
   * Subquery disebut **inner query**, sedangkan kueri utama disebut **outer query**.
2. **Penggunaan Subquery**
   * **Dalam Klausa WHERE**:
     + Menggunakan operator pembanding (<, >, =, <=, >=).
     + Dapat memanfaatkan fungsi agregat (SUM, AVG, MIN, MAX) atau operator (IN, EXISTS).
   * **Operator IN dan NOT IN**:
     + **IN**: Memeriksa apakah suatu nilai ada dalam kumpulan hasil subquery.
     + **NOT IN**: Memeriksa apakah suatu nilai tidak ada dalam kumpulan hasil subquery.

**Transaksi dalam MySQL**

1. **Definisi**
   * Transaksi adalah serangkaian operasi MySQL yang memastikan database tidak mengandung hasil operasi parsial.
   * Jika ada kegagalan, dilakukan **rollback** untuk memulihkan data. Jika berhasil, perubahan dikonfirmasi dengan **commit**.
2. **Pernyataan Transaksi MySQL**
   * **BEGIN/START TRANSACTION**: Memulai transaksi.
   * **COMMIT**: Menyimpan perubahan secara permanen.
   * **ROLLBACK**: Membatalkan perubahan.
   * **SET AUTOCOMMIT**: Mengaktifkan/menonaktifkan mode komit otomatis.
3. **Perbedaan dengan dan tanpa Transaksi**
   * Tanpa transaksi: Data dihapus permanen, tidak dapat dikembalikan.
   * Dengan transaksi: Perubahan bersifat sementara hingga dikonfirmasi, dan dapat dibatalkan dengan rollback.

**Triggering:**

* **Definisi**: Program otomatis di MySQL yang dipanggil saat terjadi **INSERT**, **UPDATE**, atau **DELETE** pada tabel.
* **Jenis**:
  1. **Row-level trigger**: Diaktifkan untuk setiap baris yang terpengaruh.
  2. **Statement-level trigger**: Dijalankan sekali untuk setiap transaksi.
* **Keuntungan**: Memeriksa integritas data, menangani kesalahan, menjalankan tugas otomatis, dan mengaudit perubahan data.
* **Kelemahan**: Validasi terbatas, debugging sulit, dan meningkatkan overhead server.

**Stored Procedure:**

* **Definisi**: Fungsi SQL yang dapat digunakan kembali dengan parameter untuk mendukung dinamika aplikasi.
* **Tujuan**: Efisiensi dalam pembuatan kode CRUD di MySQL.
* **Keuntungan**: Mengurangi lalu lintas jaringan, mendukung logika bisnis yang dapat digunakan kembali, meningkatkan keamanan.
* **Kelemahan**: Membutuhkan lebih banyak sumber daya, debugging sulit, memerlukan keahlian khusus.
* **Parameter**:
  1. **IN**: Parameter default, nilai asli tidak berubah.
  2. **OUT**: Nilai dapat diubah dan dikembalikan.
  3. **INOUT**: Kombinasi IN dan OUT.

**View Table:**

* **Definisi**: Tabel virtual/logical hasil dari query.
* **Tujuan**:
  + Mempermudah penulisan query.
  + Menyembunyikan kolom rahasia.
  + Mempercepat proses menampilkan data untuk query yang berulang.
* **Operasi**:
  + **Update** dan **hapus** view dapat dilakukan sesuai kebutuhan menggunakan sintaks tertentu.

**Temporary Table:**

* **Definisi**: Tabel sementara di MySQL yang hanya berlaku untuk satu sesi.
* **Cara Kerja**:
  + Dihapus otomatis setelah sesi berakhir.
  + Tidak muncul dalam hasil query **SHOW TABLES**.