

Modul 1: MySQL – Data Manipulation Language (DML) - INSERT

Topik

Penggunaan SQL Statement INSERT pada DBMS MySQL.

Tujuan

Setelah mempelajari modul ini, mahasiswa diharapkan dapat:

1. Memahami penggunaan SQL statement INSERT.
2. Menambahkan data ke dalam tabel menggunakan berbagai format INSERT.

Pendahuluan

DML merupakan istilah untuk beberapa sintaksis (syntax) dari SQL yang digunakan untuk melakukan perubahan pada data (isi tabel-tabel) dalam suatu database. DML terdiri dari 3 klausa utama yaitu:

1. **INSERT** : Menambah baris baru pada sebuah tabel
2. **UPDATE** : Mengubah nilai suatu baris pada sebuah tabel.
3. **DELETE** : Menghapus suatu baris dari sebuah tabel.

Pada DML terdapat dua jenis bahasa, yaitu :

1. High-Level(Non_procedural) DML.

- Digunakan secara interaktif (interpreter)
 - Dapat dijadikan satu dengan general purpose programming language (embedded)
- High-Level DML yang biasa digunakan secara interaktif disebut "Query Language".

2. Low-Level(Proedural) DML.

Digunakan secara embedded dalam suatu general purpose programming language

Bilamana kedua jenis DML diatas digunakan secara "embedded", maka : bahasa pemrograman yang digunakan disebut sebagai "Host Language" dan "DML-nya disebut "Sub Language"

Operasi INSERT

Operasi insert bertujuan untuk menyisipkan satu tuple baru ke dalam suatu relasi R.

Klausa pembentuk:

1. **INSERT**
2. **INTO**
3. **VALUES**

Format:

1. **INSERT INTO** nama_tabel (kolom1, kolom2, ...dst.) **VALUES** (nilai_kolom1, nilai_kolom2, ...dst.);
2. **INSERT INTO** nama_tabel **VALUES** (nilai_kolom1, nilai_kolom2, ...dst.); 3. [Salah satu dari kedua format sebelumnya], (nilai_kolom_kolom_baris1), (nilai_kolom_kolom_baris2), ...dst.

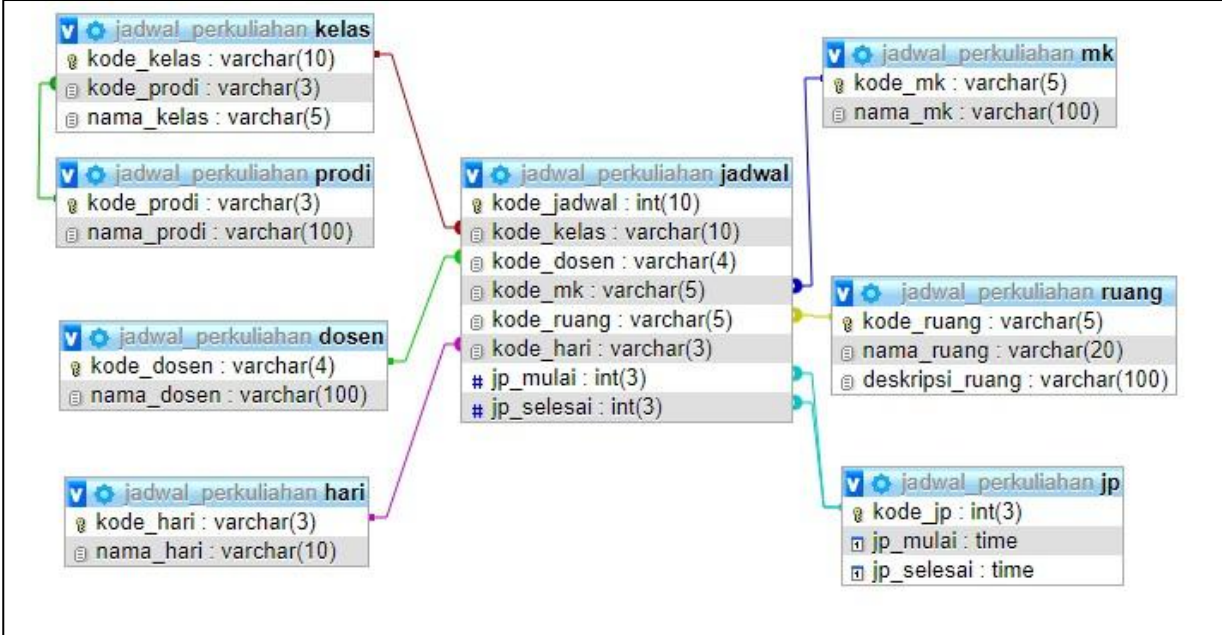
Operasi ini memungkinkan untuk melanggar empat jenis constraint sebagaimana dijelaskan berikut ini :

1. **DOMAIN Constraint** dapat dilanggar jika suatu nilai attribute yang diberikan tidak ada dalam domain yang berkorespondensi dengan attribute tadi.
2. **KEY Constraint** dapat dilanggar jika nilai key dalam tuple baru t sudah ada dalam tuple lain dalam relasi $r(R)$.
3. **ENTITY INTEGRITY** Constraint dapat dilanggar jika primary key dari tuple baru t adalah NULL
4. **REFERENTIAL INTEGRITY** Constraint dapat dilanggar jika nilai dari suatu foreign key dalam t mengacu ke suatu tuple yang tidak ada dalam relasi yang diacu.

Ada dua pilihan tindakan yang dapat dilakukan jika ada satu atau lebih constraint yang dilanggar akibat operasi insert, yaitu :

1. Menolak (reject) operasi insertion. Biasanya DBMS memberikan penjelasan mengapa proses insertion ditolak.
2. Berusaha memperbaiki alasan penolakan proses insertion. Dimana insertion akan diterima jika user melakukan perubahan nilai-nilai attribute sehingga insertion diterima.

Praktikum – Bagian 1: Percobaan Statement INSERT

Langkah	Keterangan
	<p>Perhatikan skema/model relasional/EER diagram dari database berikut.</p>  <pre> graph LR subgraph "jadwal_perkuliahan_kelas" kpk[kode_kelas : varchar(10)] kpp[kode_prodi : varchar(3)] knk[nama_kelas : varchar(5)] end subgraph "jadwal_perkuliahan_prodi" kppr[kode_prodi : varchar(3)] knpr[nama_prodi : varchar(100)] end subgraph "jadwal_perkuliahan_dosen" kpd[kode_dosen : varchar(4)] knpd[nama_dosen : varchar(100)] end subgraph "jadwal_perkuliahan_hari" kph[kode_hari : varchar(3)] knph[nama_hari : varchar(10)] end subgraph "jadwal_perkuliahan_jadwal" kjj[kode_jadwal : int(10)] kjk[kode_kelas : varchar(10)] kjd[kode_dosen : varchar(4)] kjm[kode_mk : varchar(5)] kjr[kode_ruang : varchar(5)] kjh[kode_hari : varchar(3)] jpm[jp_mulai : int(3)] jps[jp_selesai : int(3)] end subgraph "jadwal_perkuliahan_mk" km[kode_mk : varchar(5)] knm[nama_mk : varchar(100)] end subgraph "jadwal_perkuliahan_ruang" kru[kode_ruang : varchar(5)] knru[nama_ruang : varchar(20)] kdr[deskripsi_ruang : varchar(100)] end subgraph "jadwal_perkuliahan_jp" kjp[kode_jp : int(3)] jpmu[jp_mulai : time] jpse[jp_selesai : time] end kpp --> kjk kppr --> kppr kpd --> kjd kph --> kjh kjm --> km kjr --> kru kjj --> kjj jpm --> jpmu jps --> jpse </pre>
1.	<p>Untuk menambahkan data (mengisi) suatu tabel, digunakan statement (pernyataan) INSERT. Eksekusi SQL berikut untuk menambahkan 1 baris (record) baru pada tabel mk.</p> <pre>INSERT INTO mk (kode_mk, nama_mk) VALUES ('02010', 'Basis Data');</pre> <p>Statement SQL tersebut menambahkan 1 baris baru ke tabel mk pada kolom yang dinyatakan di dalam tanda kurung () pertama.</p> <p>Untuk melihat hasil SQL yang kita eksekusi tersebut, gunakan statement SELECT seperti berikut. Pembahasan lebih lengkap mengenai SELECT dijadwalkan untuk disampaikan pada pertemuan berikutnya, namun secara umum, statement SELECT digunakan untuk menyajikan record yang ada pada suatu tabel. Karakter * akan menampilkan isi dari semua kolom yang ada pada tabel.</p> <pre>SELECT * FROM mk</pre>

2.	<p>Apabila data di-insert-kan pada semua kolom tabel, maka kita dapat langsung menggunakan klausa VALUES tanpa harus menuliskan nama-nama kolom dahulu.</p> <pre>INSERT INTO mk VALUES('02041', 'Teknologi Data');</pre> <p>Statement SQL tersebut menambahkan 1 baris baru ke tabel mk pada kolom yang ada pada struktur tabel mk tanpa menyebutkan nama kolomnya.</p> <p>Untuk melihat hasil SQL yang kita eksekusi tersebut, gunakan statement SELECT seperti berikut.</p> <pre>SELECT * FROM mk</pre>
3.	<p>Untuk menambahkan beberapa kolom sekaligus dalam 1 statement digunakan statement dengan format seperti berikut.</p> <pre>INSERT INTO mk VALUES (`02004`, `Aljabar Linier`), (`02005`, `Analisis Dan Desain Berorientasi Objek`), (`02006`, `Bahasa Indonesia`);</pre> <p>Statement SQL tersebut menambahkan 3 baris baru ke tabel mk pada kolom yang ada pada struktur tabel mk tanpa menyebutkan nama kolomnya.</p> <p>Untuk melihat hasil SQL yang kita eksekusi tersebut, gunakan statement SELECT seperti berikut.</p> <pre>SELECT * FROM mk</pre>
4.	<p>Dan seperti berikut, jika hanya kolom tertentu saja yang akan diberi nilai dengan cara menyebutkan nama kolomnya.</p> <pre>INSERT INTO mk (kode_mk, nama_mk) VALUES (`02001`, `Agama`), (`02002`, `Aljabar Linier`), (`02003`, `Algoritma dan Struktur Data`);</pre>

	<p>Statement SQL tersebut menambahkan 3 baris baru ke tabel mk pada kolom yang ada pada struktur tabel mk.</p> <p>Untuk melihat hasil SQL yang kita eksekusi tersebut, gunakan statement SELECT seperti berikut.</p> <pre>SELECT * FROM mk</pre>
5.	<p>Statement INSERT juga dapat dieksekusi dengan menggunakan klausa SET alih-alih VALUES.</p> <pre>INSERT INTO mk SET Kode_mk = `02012`, nama_mk = `Digital Entrepreneurship`;</pre> <p>Statement SQL tersebut menambahkan 1 baris baru ke tabel mk pada kolom yang ada pada struktur tabel mk.</p> <p>Untuk melihat hasil SQL yang kita eksekusi tersebut, gunakan statement SELECT seperti berikut.</p> <pre>SELECT * FROM mk</pre>
6.	<p>Pada statement INSERT juga dapat digunakan klausa SELECT.</p> <p>Misalnya kita ingin menyalin semua baris pada tabel mk ke tabel mk_backup, maka kita dapat menggunakan SQL berikut. (Buat terlebih dahulu tabel “mk_backup” dengan struktur tabel yang sama dengan tabel “mk”)</p> <pre>CREATE TABLE mk_backup (kode_mk VARCHAR(5), nama_mk VARCHAR(100));</pre>
7.	<p>Kemudian masukkan data dari tabel mk ke tabel mk_backup</p> <pre>INSERT INTO mk_backup SELECT * FROM mk;</pre>
8.	<p>Setelah berhasil mengeksekusi SQL tersebut, lanjutkan ke SubTopik selanjutnya.</p>