

ORACLE PL/SQL · PERTEMUAN 03 / 14

DML – Data Manipulation Language

Mengisi dataset yang sama dan mengendalikan perubahan dengan transaksi.

Instalasi & SQL → Mini Project → PL/SQL → Final Project · Oracle XE lokal atau FreeSQL dengan login

Versi PDF materi pertemuan 03. Contoh dan tabel dapat dibaca di PDF. Kuis dapat dijalankan di halaman web: [Buka materi daring](#).

Tujuan pembelajaran

1. Mengisi dataset dengan INSERT ALL dan urutan foreign key yang benar.
2. Menggunakan INSERT, UPDATE, dan DELETE bersyarat.
3. Menetapkan keputusan COMMIT atau ROLLBACK sebagai pemilik transaksi.
4. Memakai SAVEPOINT tanpa mencampurkan DDL ke latihan rollback.

Pokok bahasan / praktikum

- INSERT
- INSERT ALL multi-data
- UPDATE bersyarat
- DELETE bersyarat
- COMMIT
- ROLLBACK
- SAVEPOINT
- Transaksi dan foreign key
- Autocommit serta pemilik transaksi

Prasyarat: Modul02 lengkap: lima tabel kosong, atau setup indeks sudah sekali dijalankan. Pilih XE lokal ATAU FreeSQL login; jangan mengisi seed dua kali.

Pilih salah satu lingkungan praktikum

Oracle XE: XE lokal dengan SQL Developer: cocok untuk instalasi, koneksi, user/schema, privilege, tablespace serta latihan SQL/PLSQL.

FreeSQL: FreeSQL dengan login: masuk menggunakan akun sendiri dan gunakan schema latihan yang diizinkan untuk SQL/PLSQL. Tidak perlu memasang XE untuk menyelesaikan latihan non-administratif.

Pengumpulan tugas menerima hasil dari salah satu lingkungan. Pastikan objek prasyarat tersedia dan tetap gunakan schema latihan, bukan schema produksi.

AMATI → PREDIKSI → JELASKAN

Mulai dari pertanyaan

1. Bagaimana memastikan UPDATE hanya mengubah produk yang dimaksud?
2. Apakah hasil SELECT setelah UPDATE berarti perubahan telah permanen?
3. Kapan membatalkan sebagian perubahan lebih tepat daripada membatalkan semuanya?

Uji pemahaman awal

1. Apa dampak UPDATE tanpa WHERE?

- Hanya primary key terkecil
- Semua baris menjadi sasaran
- Tidak ada perubahan
- Selalu menimbulkan syntax error

2. Apa yang dipertahankan ROLLBACK TO suatu savepoint?

- Semua perubahan setelahnya
- Perubahan sebelum savepoint yang belum dibatalkan
- Savepoint dari transaksi yang sudah commit
- DDL sebelumnya bisa dipulihkan

3. Seed sudah diisi lewat setup indeks. Apa langkah berikutnya?

- Ulang INSERT ALL
- DROP semua tabel
- Lewati seed dan periksa jumlah serta isi
- Ganti semua primary key

PAHAMI KONSEP · BACA TERSTRUKTUR · TERAPKAN

Teori dan pembahasan

DML mengubah isi tabel. Kebenaran perintah tidak cukup: sasaran baris, keadaan transaksi dan keputusan akhir juga harus jelas.

Konsep inti bab ini

- Periksa sasaran UPDATE/DELETE dengan SELECT dan kondisi yang sama.
- SAVEPOINT menandai posisi dalam transaksi, bukan menyimpan permanen.
- COMMIT adalah keputusan menyimpan; ROLLBACK membatalkan pekerjaan belum committed.

Urutan pembahasan

- 01 DML dan batas transaksi
- 02 Mengisi seed data secara utuh
- 03 INSERT dengan kolom dan nilai khusus
- 04 UPDATE dan kondisi yang terukur
- 05 DELETE, dependensi, dan keamanan data
- 06 SAVEPOINT dan keputusan akhir

Gambaran konsep dalam satu tabel

Perintah	Tujuan	Pemeriksaan
INSERT / INSERT ALL	Menambah baris	ID, nilai wajib dan foreign key
UPDATE	Mengubah baris yang cocok	WHERE serta jumlah baris terdampak
DELETE	Menghapus baris yang cocok	Dependensi child-parent
SAVEPOINT / ROLLBACK TO	Membatasi pemulihan sebagian	Gunakan sesi yang sama
COMMIT / ROLLBACK	Memutuskan transaksi	Pastikan hasil telah diverifikasi

01 DML dan batas transaksi

DML mengubah isi tabel, bukan definisi kolom.

INSERT menambah, **UPDATE** mengganti nilai baris yang cocok, dan **DELETE** menghapus baris yang cocok. Transaksi adalah rangkaian perubahan yang diputuskan bersama: **COMMIT** menyimpan keputusan, **ROLLBACK** membatalkan perubahan belum committed. **SELECT** pada koneksi sendiri dapat melihat perubahan sementara, sehingga tampilan hasil belum membuktikan bahwa koneksi lain sudah melihat keputusan tersebut.

POIN PEMBAHASAN

- Oracle tidak membutuhkan **BEGIN TRANSACTION** untuk latihan SQL ini.
- Perubahan DML memulai pekerjaan transaksi sesuai konteks sesi.
- Untuk praktik, nonaktifkan autocommit apabila editor menyediakan pengaturan itu.
- Pada layanan yang membatasi sesi atau mengeksekusi potongan secara terpisah, jalankan seluruh demonstrasi transaksi dalam satu eksekusi skrip agar **SAVEPOINT** dan **ROLLBACK** merujuk sesi yang sama.

02 Mengisi seed data secara utuh

Skrip seed memakai **INSERT ALL** untuk banyak baris, diikuti **SELECT 1 FROM dual** sebagai sumber satu baris.

Setiap INTO pada skrip ini menghasilkan satu sisipan untuk sumber tersebut. Ini sintaks Oracle, bukan daftar VALUES berulang bergaya produk lain. Lima kelompok entitas diisi mengikuti ketergantungan: pelanggan dan produk lebih dahulu, kemudian penjualan dan detail. Pegawai Sari dimasukkan sebelum bawahannya.

POIN PEMBAHASAN

- Seed dipakai sekali, sesudah tabel kosong.
- Apabila setup indeks sudah menyediakan data, lewati sel seed dan lakukan pemeriksaan.
- Pengulangan akan melanggar primary key, bukan menambah salinan yang sah.
- Jika gagal pada schema yang semula kosong, hentikan eksekusi, periksa pesan, lalu **ROLLBACK** perubahan seed yang belum committed sebelum memperbaiki.
- Jangan membiarkan editor terus menuju **COMMIT** setelah error yang belum dipahami.

03 INSERT dengan kolom dan nilai khusus

INSERT satu baris sebaiknya menyebut daftar kolom agar maksudnya jelas dan tidak bergantung urutan fisik kolom.

Contoh pelanggan sementara 90 memiliki email **NULL** secara eksplisit. Tanggal seed memakai literal DATE '2026-10-01', sehingga tidak bergantung format tanggal sesi. Kutip tunggal digunakan untuk teks; angka tidak perlu diberi tanda kutip atau pemisah ribuan.

POIN PEMBAHASAN

- **DEFAULT** dapat dipakai atau kolom dilewati untuk mendapatkan nilai default.
- Namun **NULL** eksplisit tetap **NULL** pada default biasa dan mungkin ditolak **NOT NULL**.

- Oracle memperlakukan string karakter kosong sebagai `NULL` ; jangan berharap email kosong mempunyai identitas tersendiri.
- Pastikan ID latihan belum digunakan.
- Dataset menyediakan pelanggan tanpa penjualan dan penjualan anonim dengan sengaja, bukan sebagai kerusakan yang harus otomatis diperbaiki.

04 UPDATE dan kondisi yang terukur

`UPDATE produk_demo SET stok = stok - 2 WHERE id_produk = 11 AND stok >= 2` mengubah maksimal satu produk karena primary key.

Kondisi stok memastikan perubahan tidak mengurangi stok ketika persediaan tidak cukup. Periksa jumlah baris yang dilaporkan editor dan `SELECT` sesudahnya. Nol baris berubah bukan selalu error SQL: bisa berarti ID tidak ada atau syarat bisnis tidak terpenuhi.

POIN PEMBAHASAN

- `UPDATE` tanpa `WHERE` mengubah semua baris tabel, termasuk jika tujuan awal hanya satu pelanggan.
- Sebelum perubahan bersyarat, jalankan `SELECT` dengan kondisi yang sama untuk memprediksi sasaran.
- Jangan mengganti harga detail masa lalu hanya karena harga katalog berubah; kedua kolom mempunyai makna berbeda.
- Batas `CHECK` masih berlaku saat `UPDATE` , sehingga nilai stok negatif ditolak oleh database.

05 DELETE, dependensi, dan keamanan data

`DELETE FROM pelanggan_demo WHERE id_pelanggan = 90` menghapus pelanggan sementara saja.

`DELETE` tanpa `WHERE` menghapus seluruh baris yang diizinkan constraint. Menghapus pelanggan yang dirujuk penjualan dapat gagal karena foreign key; DDL bersama tidak menyediakan `ON DELETE CASCADE`. Urutan child-parent diperlukan jika memang hendak menghapus data terkait, tetapi bukan tujuan eksperimen ini.

POIN PEMBAHASAN

- **DELETE** adalah DML sehingga perubahan belum committed dapat dibatalkan dalam transaksi yang sama.
- **TRUNCATE** adalah DDL dan tidak memiliki sifat rollback yang sama.
- Karena itu semua contoh transaksi memakai tabel bersama yang sudah terbentuk dan tidak menjalankan **CREATE**, **ALTER**, atau **TRUNCATE** di tengah.
- Hindari penghapusan seed untuk sekadar mencoba perintah; gunakan ID sementara dan kembalikan keadaan awal.

06 SAVEPOINT dan keputusan akhir

SAVEPOINT menandai posisi dalam transaksi.

ROLLBACK TO setelah_tambah membatalkan perubahan setelah titik itu, tetapi mempertahankan **INSERT** yang dilakukan sebelumnya. Dalam demonstrasi, penambahan pelanggan tetap terlihat, sedangkan pengurangan stok dibatalkan. Sesudah itu **DELETE** pelanggan sementara juga dibatalkan oleh **ROLLBACK** penuh, sekaligus membatalkan **INSERT** awal, sehingga data seed kembali seperti semula.

POIN PEMBAHASAN

- **COMMIT** mengakhiri transaksi, membuat perubahan permanen, dan menghapus savepoint.
- **ROLLBACK** penuh juga mengakhiri pekerjaan transaksi itu.
- Pemilik transaksi pada modul ini adalah mahasiswa melalui koneksi editor, bukan skrip tersembunyi lain.
- Jangan melakukan **COMMIT** untuk perubahan yang belum diverifikasi.
- Seed memiliki **COMMIT** akhir sebagai keputusan eksplisit; eksperimen sementara sengaja memakai **ROLLBACK** agar modul berikutnya tetap memakai data acuan yang sama.

Alur perubahan data yang aman

- 1 Baca kondisi awal
- 2 Tentukan sasaran perubahan
- 3 Jalankan DML
- 4 Periksa data dan jumlah baris
- 5 Pilih `COMMIT` atau `ROLLBACK`

Gunakan alur ini sebagai panduan saat mengikuti praktikum; bukti hasil tetap berasal dari eksekusi Oracle Anda.

AMATI DATA · IKUTI LANGKAH · PAHAM HASIL

Contoh dan tabel pembahasan

Contoh berikut dibaca langsung, tanpa parameter atau tombol simulasi. Nilai adalah **prediksi ilustratif dari data acuan**, bukan hasil eksekusi pada akun Oracle Anda.

Stok Kopi diubah lalu dibatalkan

Mulai dari seed: Kopi memiliki stok 20. Contoh dijalankan pada satu sesi dengan autocommit tidak aktif dan tanpa DDL di tengah transaksi.

Contoh SQL / PL/SQL

```
SAVEPOINT contoh_stok;  
UPDATE produk_demo SET stok = 15 WHERE id_produk = 11;  
SELECT stok FROM produk_demo WHERE id_produk = 11;  
ROLLBACK TO contoh_stok;  
SELECT stok FROM produk_demo WHERE id_produk = 11;
```

Tahap	Perintah	Stok yang terlihat pada sesi
Awal	<code>SELECT</code> stok	20
Penanda	<code>SAVEPOINT</code> contoh_stok	20
Perubahan	<code>UPDATE</code> stok = 15	15
Pemulihan	<code>ROLLBACK</code> TO contoh_stok	20

Cara membaca contoh

1. Baca stok sebelum mengubahnya. Pastikan kondisi `WHERE` menunjuk id_produk 11.
2. Buat savepoint lalu ubah stok. `SELECT` di sesi yang sama akan melihat nilai kerja yang baru.
3. `ROLLBACK` TO memulihkan perubahan sesudah savepoint; tidak mengakhiri seluruh transaksi.

Yang perlu dipahami: Jika memilih `COMMIT` setelah `UPDATE`, nilai baru menjadi tersimpan. `ROLLBACK` tidak dapat membatalkan commit yang sudah selesai.

Video pembelajaran

Video utama — fokus bab ini

`COMMIT`, `ROLLBACK`, `SAVEPOINT` dan transaksi Oracle; video tambahan menjelaskan `INSERT`, `UPDATE` dan `DELETE`.

Oracle Database Transactions | Oracle Transactions | COMMIT, ROLLBACK, SAVEPOINT | Oracle Tutorial

RK SoftAcademy · [Buka video utama di YouTube](#)

Video tambahan sesuai subtopik

1. [Oracle DB Programming Section 12: DML \(INSERT, UPDATE, DELETE, MERGE\)](#)

Eric Cameron

Fokus: Oracle DML: INSERT, UPDATE, DELETE dan pengantar MERGE.

MERGE merupakan tambahan; tugas utama tetap INSERT ALL, perubahan bersyarat dan kontrol transaksi.

Batas cakupan dan panduan menonton: Latihan INSERT ALL mengikuti script Oracle di halaman. Jalankan eksperimen transaksi dalam sesi yang sama dengan autocommit diperiksa.

Video merupakan pendamping, bukan pengganti keseluruhan bab. Pemeriksaan relevansi memakai transkrip yang tersedia, deskripsi pembuat dan bab waktu; bukan klaim seluruh durasi telah ditonton. Versi/editor atau bahasa dapat berbeda. Praktikum tetap mengikuti SQL/PLSQL Oracle pada halaman dan pilihan XE atau FreeSQL login. Jika pemutar dibatasi oleh YouTube, gunakan tautan Buka video.

Praktikum Oracle XE atau FreeSQL

Pilih salah satu lingkungan praktikum

Oracle XE: XE lokal dengan SQL Developer: cocok untuk instalasi, koneksi, user/schema, privilege, tablespace serta latihan SQL/PLSQL.

FreeSQL: FreeSQL dengan login: masuk menggunakan akun sendiri dan gunakan schema latihan yang diizinkan untuk SQL/PLSQL. Tidak perlu memasang XE untuk menyelesaikan latihan non-administratif.

Pengumpulan tugas menerima hasil dari salah satu lingkungan. Pastikan objek prasyarat tersedia dan tetap gunakan schema latihan, bukan schema produksi.

Gunakan [FreeSQL dengan login](#) atau SQL Developer yang terhubung ke XE. [Baca panduan dua jalur praktikum](#). Jalankan kode sesuai urutan dan keterangan tiap langkah.

Output contoh adalah prediksi berdasarkan dataset, bukan bukti bahwa kode telah dijalankan pada akun Anda. Periksa koneksi, hak membuat objek, autocommit dan panel DBMS_OUTPUT. Slash pada baris sendiri adalah pemisah mode skrip klien tertentu,

bukan statement PL/SQL.

Langkah 1 — Seed lengkap bersama: hanya pada lima tabel kosong

```

INSERT ALL
  INTO pelanggan_demo VALUES (1, 'Ayu', 'Subang', 'ayu@example.test')
  INTO pelanggan_demo VALUES
(2, 'Bima', 'Bandung', 'bima@example.test')
  INTO pelanggan_demo VALUES (3, 'Citra', NULL, NULL)
SELECT 1 FROM dual;
INSERT ALL
  INTO produk_demo VALUES (11, 'Kopi', 30000, 20)
  INTO produk_demo VALUES (12, 'Teh', 15000, 30)
  INTO produk_demo VALUES (13, 'Gula', 12000, 40)
  INTO produk_demo VALUES (14, 'Sabun', 10000, 0)
SELECT 1 FROM dual;
INSERT ALL
  INTO penjualan_demo VALUES (101, 1, DATE '2026-10-01', 'LUNAS')
  INTO penjualan_demo VALUES (102, 2, DATE '2026-10-02', 'BARU')
  INTO penjualan_demo VALUES (103, NULL, DATE '2026-10-03', 'BARU')
SELECT 1 FROM dual;
INSERT ALL
  INTO detail_jual_demo VALUES (101, 11, 10, 30000)
  INTO detail_jual_demo VALUES (102, 12, 10, 15000)
  INTO detail_jual_demo VALUES (103, 13, 5, 12000)
SELECT 1 FROM dual;
INSERT INTO pegawai_demo VALUES (1, 'Sari', NULL);
INSERT INTO pegawai_demo VALUES (2, 'Rudi', 1);
INSERT INTO pegawai_demo VALUES (3, 'Tono', 1);
COMMIT;

```

Persis INSERT ALL dan INSERT pada setup.json, termasuk COMMIT akhir. Jika setup telah dijalankan, lewati sel ini. Hentikan dan rollback jika ada error sebelum melanjutkan.

Langkah 2 — Periksa jumlah data acuan

```

SELECT 'pelanggan' AS entitas, COUNT(*) AS jumlah FROM
  pelanggan_demo

```

```

UNION ALL SELECT 'produk', COUNT(*) FROM produk_demo
UNION ALL SELECT 'penjualan', COUNT(*) FROM penjualan_demo
UNION ALL SELECT 'detail', COUNT(*) FROM detail_jual_demo
UNION ALL SELECT 'pegawai', COUNT(*) FROM pegawai_demo;

```

Diprediksi masing-masing 3, 4, 3, 3, 3 pada seed utuh. Ini prediksi dari data, bukan hasil eksekusi Oracle penyusun.

Langkah 3 — Satu skrip untuk perubahan dan pembatalan

```

-- Pastikan ID 90 belum dipakai; seluruh skrip satu sesi,
autocommit off.
SELECT * FROM pelanggan_demo WHERE id_pelanggan = 90;
INSERT INTO pelanggan_demo (id_pelanggan,nama,kota,email)
VALUES (90,'Latihan sementara','Subang',NULL);
SAVEPOINT setelah_tambah;
SELECT id_produk, stok FROM produk_demo WHERE id_produk = 11 AND
stok >= 2;
UPDATE produk_demo SET stok = stok - 2 WHERE id_produk = 11 AND
stok >= 2;
SELECT id_produk, stok FROM produk_demo WHERE id_produk = 11;
ROLLBACK TO setelah_tambah;
SELECT id_produk, stok FROM produk_demo WHERE id_produk = 11;
DELETE FROM pelanggan_demo WHERE id_pelanggan = 90;
SELECT * FROM pelanggan_demo WHERE id_pelanggan = 90;
ROLLBACK;
SELECT * FROM pelanggan_demo WHERE id_pelanggan = 90;
SELECT id_produk, stok FROM produk_demo WHERE id_produk = 11;

```

Seed stok Kopi 20 diprediksi menjadi 18, lalu kembali 20. Rollback penuh membatalkan INSERT dan DELETE pelanggan 90. Jangan jalankan bila ID 90 sudah dimiliki data lain.

Tugas / latihan

1. Gunakan XE lokal ATAU FreeSQL login; isi seed sekali bila diperlukan dan laporkan jumlah setiap tabel.
2. Jalankan satu skrip transaksi pada sesi yang sama, dokumentasikan stok sebelum/sesudah rollback dan pelanggan sementara.

3. Tulis perubahan status penjualan 102 menjadi LUNAS dengan WHERE, periksa sasaran, kemudian ROLLBACK agar seed tetap BARU; jelaskan keputusan COMMIT yang akan dipakai jika perubahan memang disetujui.

Pertanyaan analisis

1. Bedakan nol baris terdampak, kegagalan constraint, dan perubahan yang belum committed.
2. Mengapa COMMIT tidak boleh disisipkan sebelum Anda memilih antara menerima atau membatalkan perubahan?

Pengumpulan

Tugas boleh dikerjakan di **Oracle XE lokal atau FreeSQL dengan login**; cukup pilih salah satu. Kumpulkan satu laporan melalui LMS/kanal yang ditetapkan dosen. Cantumkan Nama, NPM, pertemuan dan lingkungan yang dipakai. Sertakan kode SQL/PLSQL sebagai teks, output aktual atau pesan error, serta jawaban analisis.

Jika FreeSQL menyediakan tautan berbagi yang dapat diakses dosen, tautan boleh ditambahkan sebagai bukti pendamping, bukan pengganti hasil eksekusi yang terdokumentasi. Jangan membagikan password, token atau koneksi pribadi.

Hasil yang belum dieksekusi diberi label “prediksi”. Screenshot ilustrasi browser bukan bukti praktikum Oracle.

Rubrik penilaian

- Ketepatan konsep dan rancangan: **20%**
- Kebenaran query/program: **30%**
- Pengujian normal, batas dan gagal: **20%**
- Analisis serta penjelasan hasil: **20%**
- Identitas dan kerapian laporan: **10%**

Referensi resmi

- [Oracle Database 19c SQL Language Reference: INSERT](#)
- [Oracle Database 19c SQL Language Reference: UPDATE](#)
- [Oracle Database 19c SQL Language Reference: DELETE](#)

- [Oracle Database 19c SQL Language Reference: COMMIT](#)
- [Oracle Database 19c SQL Language Reference: ROLLBACK](#)
- [Oracle Database 19c SQL Language Reference: SAVEPOINT](#)

Panduan XE21c digunakan untuk konteks instalasi lokal; dokumentasi SQL/PLSQL untuk konsep bahasa. Versi FreeSQL/editor dapat berbeda, sehingga periksa fitur dan pesan aktual.
