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

BELAJAR BERURUTAN · SATU PERUBAHAN · SATU PEMERIKSAAN

Teori dan pembahasan

Bab ini dipelajari dari keadaan data yang sudah ada, kemudian menambah, mengubah, menghapus, dan akhirnya memilih apakah perubahan diterima atau dibatalkan. Setiap contoh menyebut prasyarat, kode, prediksi hasil, dan cara membacanya.

Sebelum mulai

1. Gunakan tabel bersama dari Pertemuan 02 pada schema latihan pribadi di Oracle XE atau FreeSQL dengan login.
2. Pastikan Anda tahu apakah seed sudah terisi. Jangan mengulang INSERT seed jika datanya sudah ada.
3. Gunakan sesi latihan tanpa perubahan lain yang belum disimpan; ROLLBACK penuh juga membatalkan pekerjaan lain dalam transaksi yang sama.
4. Nonaktifkan autocommit bila tersedia. Untuk demonstrasi transaksi, jalankan satu blok lengkap pada sesi yang sama.
5. Contoh bertanda ID 90, 91, atau 92 hanya boleh dijalankan jika ID tersebut belum dipakai. Jangan menghapus data lain untuk memenuhi contoh.

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 belajar

01 DML dan persiapan latihan

02 SELECT: membaca keadaan awal

03 INSERT: menambahkan satu baris

04 INSERT ALL: memahami pengisian seed

05 UPDATE: mengubah baris yang dipilih

06 DELETE: menghapus baris dengan aman

07 COMMIT dan ROLLBACK: memilih hasil akhir

08 SAVEPOINT: membatalkan sebagian perubahan

09 Mengikuti praktikum dari awal sampai akhir

10 Mengenali kesalahan dan memeriksa pemahaman

01 DML dan persiapan latihan

Mulailah dengan dua pertanyaan: tabel apa yang akan diubah, dan baris mana yang menjadi sasaran? DML berkaitan dengan isi data. Perintah pengendalian transaksi dipelajari bersama DML karena menentukan nasib perubahan, tetapi fungsinya berbeda.

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.

Kelompok	Perintah pada bab ini	Peran
DML	INSERT, UPDATE, DELETE	Menambah, mengubah, dan menghapus isi tabel.
Pembacaan data	SELECT	Memeriksa keadaan awal dan hasil perubahan.
Pengendalian transaksi	COMMIT, ROLLBACK, SAVEPOINT	Menyimpan keputusan atau membatalkan pekerjaan yang belum committed.
DDL dari bab sebelumnya	CREATE, ALTER, TRUNCATE	Mengubah objek/struktur; jangan dicampur dalam eksperimen rollback.

Untuk pertama kali, gunakan produk 11 (Kopi) dengan stok seed 20 sebagai objek pengamatan. Pelanggan 90 dipakai sebagai data sementara. Tabel pelanggan dan produk adalah induk; penjualan dan detail menyimpan hubungan yang memerlukan perhatian terhadap foreign key.

02 SELECT: membaca keadaan awal

Sebelum menulis perubahan, baca data yang akan menjadi sasaran.

SELECT berperan sebagai jendela pengamatan. Daftar setelah SELECT menunjukkan kolom yang ingin dilihat, FROM menunjukkan tabel, dan WHERE membatasi baris. Anda belum perlu mempelajari seluruh variasi query untuk melakukan pemeriksaan dasar ini.

```
SELECT id_produk, nama_produk, harga, stok
FROM produk_demo
WHERE id_produk = 11;
SELECT id_pelanggan, nama FROM pelanggan_demo
WHERE id_pelanggan IN (90, 91, 92);
```

Pemeriksaan	Prediksi jika seed sesuai	Tindakan sebelum lanjut
Produk 11	Kopi; harga 30000; stok 20	Jika berbeda, catat keadaan sebenarnya dan periksa lingkungan latihan.
Pelanggan 90, 91, 92	Tidak ditemukan baris	Jika sudah ada, jangan jalankan contoh dengan ID tersebut.

Simpan hasil awal sebagai pembanding. Jika UPDATE nanti memakai WHERE id_produk = 11 AND stok >= 2, ubah SELECT pemeriksaan agar memakai kondisi itu juga. SELECT awal membantu memahami sasaran, tetapi tidak membekukan data terhadap perubahan oleh sesi lain; karena itu tetap periksa hasil dan jumlah baris sesudah DML.

Dalam materi ini, angka hasil disebut prediksi. Output sebenarnya harus diperoleh mahasiswa pada lingkungan Oracle yang dipakai, bukan disalin dari tabel penjelasan.

---

03 INSERT: menambahkan satu baris

Pelajari INSERT satu baris terlebih dahulu. Polanya adalah `INSERT INTO nama_tabel (kolom1, kolom2) VALUES (nilai1, nilai2)`. Daftar kolom dan daftar nilai harus mempunyai jumlah yang sama. Bedakan teks, angka, tanggal, dan NULL sebelum menjalankan perintah.

`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.

Contoh bertahap — tambah satu pelanggan sementara

Prasyarat: tabel sudah berisi seed, ID 90 belum ada, autocommit tidak aktif, dan sesi tidak menyimpan pekerjaan lain. Jalankan satu blok lengkap berikut pada schema latihan sendiri.

```
SELECT id_pelanggan, nama FROM pelanggan_demo WHERE id_pelanggan
= 90;
INSERT INTO pelanggan_demo (id_pelanggan, nama, kota, email)
VALUES (90, 'Latihan sementara', 'Subang', NULL);
SELECT id_pelanggan, nama, kota, email
FROM pelanggan_demo WHERE id_pelanggan = 90;
ROLLBACK;
SELECT id_pelanggan, nama FROM pelanggan_demo WHERE id_pelanggan
= 90;
```

Prediksi, bukan rekaman eksekusi Oracle: hasil berikut dihitung dari data seed dan urutan perintah. Bandingkan dengan output aktual Anda; bila kondisi awal berbeda, hasil dapat berbeda.

Tahap	Prediksi hasil	Alasan
Sebelum INSERT	0 baris untuk ID 90	ID latihan harus belum dipakai.

Tahap	Prediksi hasil	Alasan
Sesudah INSERT	1 baris: 90, Latihan sementara, Subang, NULL	Nilai dipasangkan dengan urutan kolom yang ditulis.
Sesudah ROLLBACK	0 baris untuk ID 90	Penambahan sementara dibatalkan.

Cara membaca: Baca pasangan kolom dan nilainya satu per satu: id_pelanggan menerima 90, nama menerima teks, kota menerima Subang, dan email menerima NULL tanpa tanda kutip. Nomor kolom dan jumlah nilai harus cocok. INSERT yang berhasil belum otomatis menjadi perubahan permanen.

Nilai khusus yang sering membingungkan: NULL menyatakan tidak adanya nilai, bukan teks 'NULL'. Untuk mencari nilai yang tidak terisi, gunakan `IS NULL`, bukan menyamakan dengan `= NULL`. Pada kolom stok produk, default 0 dipakai bila kolom dilewati atau DEFAULT diminta; aturan NOT NULL tetap harus dipatuhi.

---

04 INSERT ALL: memahami pengisian seed

Setelah memahami satu pasangan kolom–nilai, barulah pelajari penambahan beberapa baris. Bedakan contoh dua pelanggan sementara di bawah dengan proses pengisian dataset acuan. Seed adalah keadaan awal yang sengaja disamakan agar hasil antarpertemuan dapat dibandingkan.

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.

Contoh bertahap — menambah dua baris dengan INSERT ALL

Ini contoh terpisah, bukan perintah untuk mengulang seluruh seed. Pastikan ID 91 dan 92 belum dipakai. Jika salah satu sudah ada, jangan jalankan blok ini.

```
SELECT id_pelanggan FROM pelanggan_demo WHERE id_pelanggan IN
(91, 92);
INSERT ALL
 INTO pelanggan_demo (id_pelanggan, nama, kota, email)
 VALUES (91, 'Latihan A', 'Subang', NULL)
 INTO pelanggan_demo (id_pelanggan, nama, kota, email)
 VALUES (92, 'Latihan B', 'Bandung', NULL)
SELECT 1 FROM dual;
SELECT id_pelanggan, nama FROM pelanggan_demo
WHERE id_pelanggan IN (91, 92) ORDER BY id_pelanggan;
ROLLBACK;
```

Prediksi, bukan rekaman eksekusi Oracle: hasil berikut dihitung dari data seed dan urutan perintah. Bandingkan dengan output aktual Anda; bila kondisi awal berbeda, hasil dapat berbeda.

Bagian	Cara membaca
INSERT ALL	Mulai satu pernyataan untuk beberapa sisipan.
INTO pertama	Siapkan baris pelanggan 91.
INTO kedua	Siapkan baris pelanggan 92.
SELECT 1 FROM dual	Sumber satu baris; pada contoh ini setiap INTO dijalankan sekali.

Bagian	Cara membaca
SELECT pemeriksaan	Diprediksi terlihat dua pelanggan sementara sebelum ROLLBACK.
ROLLBACK	Batalkan kedua pelanggan sementara.

Cara membaca: Titik koma penutup pernyataan INSERT ALL berada setelah SELECT 1 FROM dual, bukan setelah setiap INTO. Jangan meniru sintaks banyak VALUES dari produk database lain tanpa memeriksa dukungan versi. Skrip seed lengkap yang benar tetap tersedia pada Praktikum Langkah 1; jalankan hanya jika tabel masih kosong.

---

05 UPDATE: mengubah baris yang dipilih

Pada UPDATE, urutan membacanya adalah tabel sasaran → nilai baru pada SET → baris yang dipilih oleh WHERE. Menulis SET stok = stok - 2 berbeda dengan SET stok = 2. Bentuk pertama memakai nilai lama; bentuk kedua menggantinya dengan angka tetap.

`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.

Contoh bertahap — mengurangi stok Kopi tanpa menyentuh produk lain

Prasyarat khusus contoh: produk 11 ada dengan stok seed 20. Jika keadaan awal berbeda, hentikan dan periksa konteks; jangan menimpa stok agar sama dengan buku. Contoh ini berdiri sendiri dan berakhir dengan ROLLBACK.

```
SELECT id_produk, nama_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;
SELECT id_produk, stok FROM produk_demo WHERE id_produk = 11;
```

Prediksi, bukan rekaman eksekusi Oracle: hasil berikut dihitung dari data seed dan urutan perintah. Bandingkan dengan output aktual Anda; bila kondisi awal berbeda, hasil dapat berbeda.

Tahap	Prediksi stok produk 11	Makna
SELECT awal	20	Satu baris memenuhi ID dan syarat stok.
Sesudah UPDATE	18	Stok lama dikurangi 2, bukan diganti menjadi 2.
Sesudah ROLLBACK	20	Perubahan dalam transaksi latihan dibatalkan.

Cara membaca: Baca perintah dari kiri ke kanan: UPDATE menentukan tabel, SET menentukan nilai baru, dan WHERE menentukan sasaran. Gunakan kondisi WHERE yang sama pada pemeriksaan awal dan perintah perubahan. Syarat stok >= 2 melindungi aturan contoh, tetapi tidak menggantikan pemeriksaan jumlah baris terdampak.

Jangan menyimpulkan hanya dari tidak munculnya pesan error. Jika ID tidak ditemukan atau stok tidak memenuhi syarat, UPDATE dapat selesai tanpa mengubah satu baris pun. Pisahkan tiga keadaan: satu baris berubah sesuai tujuan, nol baris memenuhi kondisi, atau perintah gagal karena aturan/constraint.

06 DELETE: menghapus baris dengan aman

DELETE memakai FROM untuk menentukan tabel dan WHERE untuk membatasi baris yang dihapus. SELECT dengan kondisi yang sama perlu dilakukan sebelum penghapusan. Untuk latihan pemula, pilih data sementara yang memang dibuat untuk dibatalkan, bukan data induk seed yang sedang dirujuk tabel lain.

`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.

Contoh bertahap — hapus hanya baris sementara

Pastikan ID 90 belum ada sebelum menjalankan blok. Kita membuat baris sementara, memeriksanya, menghapusnya, lalu membatalkan seluruh transaksi contoh. Data seed tidak dihapus.

```
INSERT INTO pelanggan_demo (id_pelanggan, nama, kota, email)
VALUES (90, 'Latihan sementara', 'Subang', NULL);
SELECT id_pelanggan, nama FROM pelanggan_demo WHERE id_pelanggan
= 90;
DELETE FROM pelanggan_demo WHERE id_pelanggan = 90;
SELECT id_pelanggan, nama FROM pelanggan_demo WHERE id_pelanggan
= 90;
```

```
ROLLBACK;
SELECT id_pelanggan, nama FROM pelanggan_demo WHERE id_pelanggan
= 90;
```

Prediksi, bukan rekaman eksekusi Oracle: hasil berikut dihitung dari data seed dan urutan perintah. Bandingkan dengan output aktual Anda; bila kondisi awal berbeda, hasil dapat berbeda.

Tahap	Prediksi baris ID 90	Penjelasan
Setelah INSERT	1 baris	Baris hanya ditambahkan dalam transaksi latihan.
Setelah DELETE	0 baris	WHERE memilih pelanggan sementara 90.
Setelah ROLLBACK penuh	0 baris	ROLLBACK membatalkan INSERT dan DELETE sekaligus; keadaan awal memang belum memiliki ID 90.

Cara membaca: DELETE menghapus baris, bukan menghapus struktur tabel. Jangan mencoba DELETE pada pelanggan 1 atau 2 sekadar untuk melihat efeknya karena mereka memiliki transaksi terkait. Untuk melihat baris muncul kembali setelah membatalkan DELETE saja, diperlukan SAVEPOINT sebelum DELETE; konsep pemulihan sebagian dipelajari pada bagian 08.

07 COMMIT dan ROLLBACK: memilih hasil akhir

Berhasil menjalankan DML belum sama dengan memutuskan menyimpan perubahan.

Transaksi memberi kesempatan memeriksa serangkaian perubahan sebelum diputuskan bersama. SELECT pada sesi sendiri dapat melihat hasil kerja yang belum committed. Karena itu melihat stok 18 pada layar bukan bukti bahwa perubahan telah menjadi keputusan permanen. Pertanyaan akhir yang perlu dijawab adalah: apakah hasil ini benar dan memang akan diterima?

Perintah	Yang terjadi	Yang tidak boleh diasumsikan
COMMIT	Mengakhiri transaksi, menerima perubahan, menghapus savepoint, dan melepaskan kunci transaksi.	ROLLBACK sesudahnya tidak dapat mengurungkan commit yang sudah selesai.
ROLLBACK	Mengakhiri transaksi dan membatalkan perubahan yang belum committed.	Bukan pembatalan data dari transaksi lama yang sudah di-commit.
ROLLBACK TO nama_savepoint	Membatalkan perubahan setelah penanda tertentu.	Tidak mengakhiri seluruh transaksi dan tidak menyimpan permanen bagian sebelumnya.

Contoh pada teori sengaja diakhiri ROLLBACK agar dataset acuan tetap utuh. COMMIT dipakai ketika penambahan/perubahan benar-benar disetujui, misalnya saat mengisi seed untuk pertama kali melalui skrip yang telah diperiksa. Jangan menambahkan COMMIT untuk menghilangkan rasa ragu terhadap error; justru pastikan error dan keadaan transaksi telah dipahami.

Dalam Oracle, DDL dapat memicu commit implisit. Akibatnya, menyisipkan CREATE, ALTER, atau TRUNCATE di tengah eksperimen bukan cara yang aman untuk mempertahankan kemampuan rollback. Pisahkan pembuatan tabel pada bab/setup yang sesuai dari eksperimen DML ini. Menutup aplikasi juga jangan dijadikan cara memilih hasil transaksi; gunakan keputusan COMMIT atau ROLLBACK yang eksplisit.

Penjelasan ini mengikuti dokumentasi COMMIT dan ROLLBACK Oracle pada bagian referensi. Tugas mahasiswa tetap memerlukan output aktual pada sesi latihan sendiri.

---

08 SAVEPOINT: membatalkan sebagian perubahan

Gunakan SAVEPOINT setelah memahami pembatalan penuh. Bayangkan transaksi sebagai rangkaian langkah dan savepoint sebagai penanda di antara langkah-langkah itu. Penanda tidak membuat perubahan sebelumnya permanen; ia hanya memberi sasaran pemulihan sebagian.

`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.

Contoh bertahap — bedakan ROLLBACK TO dan ROLLBACK penuh

Prasyarat: ID 90 belum ada, stok produk 11 masih 20, autocommit tidak aktif, tidak ada pekerjaan lain yang belum diputuskan. Jalankan seluruh blok dalam satu sesi/eksekusi skrip.

```
INSERT INTO pelanggan_demo (id_pelanggan, nama, kota, email)
VALUES (90, 'Latihan sementara', 'Subang', NULL);
SAVEPOINT setelah_tambah;
UPDATE produk_demo SET stok = stok - 2
WHERE id_produk = 11 AND stok >= 2;
SELECT stok FROM produk_demo WHERE id_produk = 11;
ROLLBACK TO setelah_tambah;
SELECT stok FROM produk_demo WHERE id_produk = 11;
SELECT id_pelanggan FROM pelanggan_demo WHERE id_pelanggan = 90;
ROLLBACK;
SELECT id_pelanggan FROM pelanggan_demo WHERE id_pelanggan = 90;
```

Prediksi, bukan rekaman eksekusi Oracle: hasil berikut dihitung dari data seed dan urutan perintah. Bandingkan dengan output aktual Anda; bila kondisi awal berbeda, hasil dapat berbeda.

Tahap	Stok Kopi	Baris pelanggan 90
Keadaan awal	20	Belum ada
INSERT lalu SAVEPOINT	20	Ada, belum committed
UPDATE stok	18	Tetap ada
ROLLBACK TO setelah_tambah	20	Tetap ada karena INSERT sebelum savepoint
ROLLBACK penuh	20	Tidak ada; seluruh perubahan transaksi dibatalkan

Cara membaca: ROLLBACK TO tidak berarti kembali ke keadaan awal sesi. Ia hanya membatalkan perubahan sesudah penanda yang dipilih dan tidak mengakhiri seluruh transaksi. Sebaliknya ROLLBACK penuh membatalkan seluruh pekerjaan yang belum committed. COMMIT yang dilakukan sebelumnya tidak dapat diurungkan oleh keduanya.

09 Mengikuti praktikum dari awal sampai akhir

Sesudah setiap konsep dipahami, gunakan satu skrip transaksi yang utuh, bukan menggabungkan potongan contoh secara sembarang.

Contoh INSERT, INSERT ALL, UPDATE, DELETE, dan SAVEPOINT di atas adalah demonstrasi terpisah dengan keadaan awal yang dinyatakan. Masing-masing mempunyai titik akhir pembatalan. Praktikum di bawah merangkai ide-ide tersebut dengan pelanggan sementara 90 dan stok produk 11. Baca keseluruhan skrip sebelum menjalankannya agar Anda mengetahui kapan data sementara ada dan kapan stok berubah.

Urutan	Yang dikerjakan	Yang diperiksa
1	Cek keberadaan tabel dan seed	Gunakan hasil Praktikum Langkah 2; jangan mengulang seed yang sudah terisi.
2	Cek ID 90 dan stok awal	ID 90 belum ada; stok acuan Kopi 20.
3	INSERT pelanggan sementara	Ada satu pelanggan 90 dalam transaksi kerja.
4	SAVEPOINT setelah_tambah	INSERT berada sebelum penanda.
5	UPDATE stok dengan WHERE	Stok diprediksi 18 bila satu baris memenuhi kondisi.
6	ROLLBACK TO setelah_tambah	Stok kembali 20, pelanggan sementara masih ada.
7	DELETE pelanggan sementara	ID 90 tidak terlihat pada keadaan kerja saat itu.
8	ROLLBACK penuh dan SELECT akhir	Data acuan kembali: ID 90 tidak ada, stok tetap 20.

Pada setiap SELECT, tulis prediksi terlebih dahulu, lalu cocokkan dengan output aktual. Jika berbeda, jangan meneruskan hanya karena langkah berikutnya tersedia. Periksa kondisi awal, sesi yang dipakai, autocommit, dan apakah perintah sebelumnya gagal. Jika eksperimen dihentikan, batalkan hanya transaksi latihan yang memang Anda miliki.

Contoh stok 20 → 15 → 20 pada bagian Contoh dan tabel pembahasan adalah ilustrasi tambahan yang berdiri sendiri. Praktikum utama memakai pengurangan 2 sehingga 20 → 18 → 20. Angka berbeda tidak berarti aturan transaksi berubah; perhatikan perintah SET yang digunakan pada masing-masing contoh.

---

10 Mengenali kesalahan dan memeriksa pemahaman

Belajar DML berarti mampu menjelaskan sasaran, hasil, dan keputusan transaksi, bukan sekadar menghasilkan perintah yang bisa diketik.

Gejala	Kemungkinan penyebab	Langkah pemeriksaan
INSERT ditolak	ID sudah dipakai, nilai wajib kosong, nilai unik bentrok, atau referensi induk tidak ada.	Baca pesan aktual, periksa nilai dan hubungan tabel; jangan mengulang seed secara membabi buta.
UPDATE/DELETE berdampak nol baris	WHERE tidak menemukan baris atau syarat bisnis tidak terpenuhi.	Jalankan SELECT dengan kondisi sama, lalu jelaskan mengapa tidak cocok.
Banyak baris berubah padahal target satu	WHERE hilang atau terlalu luas.	Jangan COMMIT. Pada sesi latihan yang sesuai, ROLLBACK dan perbaiki sasaran.
Stok tidak kembali sesuai prediksi	Ada commit, autocommit, DDL, sesi berbeda, atau kondisi awal berbeda.	Periksa urutan skrip dan pengaturan lingkungan; jangan menimpa data untuk meniru tabel contoh.
SAVEPOINT tidak dapat dipakai	Penanda tidak tersedia pada transaksi/sesi aktif.	Periksa nama penanda dan apakah ada COMMIT, ROLLBACK penuh, atau pergantian sesi.

Uji pemahaman sebelum tugas

1. Jelaskan mengapa INSERT sebaiknya menyebut kolom, dan cocokkan setiap nilai dengan kolom tujuannya.
2. Jelaskan perbedaan SET stok = stok - 2 dengan SET stok = 2.
3. Tentukan baris sasaran SELECT, UPDATE, dan DELETE dari kondisi WHERE yang sama.
4. Terangkan mengapa pelanggan 90 masih ada setelah ROLLBACK TO setelah_tambah tetapi hilang setelah ROLLBACK penuh.
5. Terangkan mengapa COMMIT tidak boleh disisipkan sebelum memilih menerima atau membatalkan perubahan.

Jika lima pertanyaan itu dapat dijawab dengan merujuk keadaan sebelum dan sesudah perintah, lanjutkan ke praktikum dan tugas. Dokumentasikan hasil yang benar-benar diperoleh; jangan menuliskan prediksi atau ilustrasi browser sebagai hasil eksekusi Oracle.

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 · PAHAMI 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	`SELECT` stok	20
Penanda	`SAVEPOINT` contoh_stok	20
Perubahan	`UPDATE` stok = 15	15
Pemulihan	`ROLLBACK` 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.
