POLITEKNIK ELEKTRONIKA NEGERI SURABAYA

DIKTISAINTEK BERDAMPAK KAMPUS INOVASI



Mengelola Skema, Data, dan Concurrency

Workshop Administrasi Basis Data

Yunia Ikawati

Teknik Informatika-PENS

OVERVIEW

01
Skema dalam Oracle Database

02
Mengelola Skema

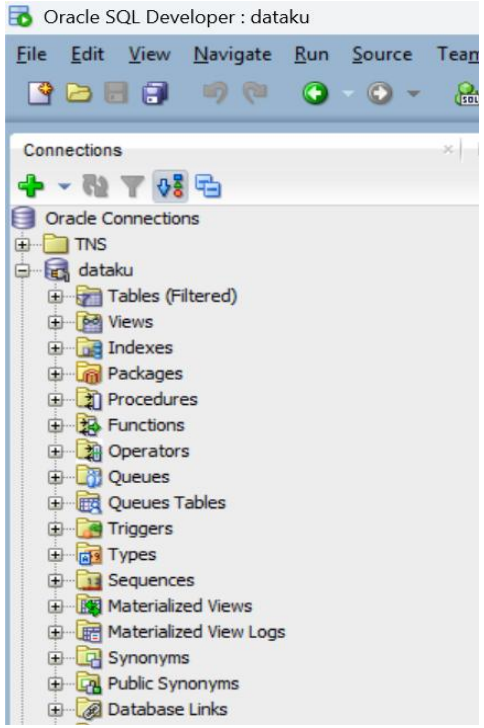
03
Pengelolaan Data pada Oracle

04
Konsep Concurrency dalam Oracle

05
Manajemen Concurrency

SKEMA DALAM ORACLE DATABASE

- **Skema adalah** kumpulan objek database (tabel, indeks, prosedur, trigger dll.) yang dimiliki oleh pengguna/user.
- **Komponen Skema:**
  - ✓ **Tabel** (Menyimpan data dalam format baris dan kolom.)
  - ✓ **View** (Representasi virtual dari data dalam tabel)
  - ✓ **Indeks** (Meningkatkan kecepatan akses data)
  - ✓ **Prosedur** (Menjalankan operasi tertentu)
  - ✓ **Fungsi** (Memproses atau mengolah data dalam database)
  - ✓ **Trigger** (Program yang berjalan otomatis Ketika terjadi aksi tertentu pada tabel)



Contoh Skema Database

MENGELOLA SKEMA

A. Membuat Skema

- Perintah CREATE USER untuk pengguna.
- Hak akses dengan GRANT.

```
CREATE USER username IDENTIFIED BY password;
```

- `username` : Nama pengguna yang akan dibuat.
- `password` : Kata sandi untuk pengguna.

```
GRANT CREATE SESSION TO username;
GRANT CREATE TABLE, CREATE VIEW, CREATE PROCEDURE, CREATE SEQUENCE TO usern
GRANT ALL PRIVILEGES TO username;
```

B. Mengelola Objek Skema:

- Perintah CREATE, ALTER, DROP.
- Backup Objek (Menggunakan Data Pump Export (EXPDP))
- Restore Objek (Menggunakan Data Pump Import (IMPDP))

```
CREATE USER schema_name IDENTIFIED BY password;

ALTER USER schema_name IDENTIFIED BY new_password;
DROP USER schema_name CASCADE;
```

PENGELOLAAN DATA PADA ORACLE

1. **Operasi Dasar:**
   - ✓ Insert,
   - ✓ Update,
   - ✓ Delete

```
INSERT INTO table_name (column1, column2, ...)
VALUES (value1, value2, ...);
```

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

```
DELETE FROM table_name
WHERE condition;
```

2. **Manipulasi Data:**
   - ✓ Perintah SELECT untuk mengambil data.
   - ✓ Menambahkan filter dengan WHERE.

```
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

KONSEP CONCURRENCY DALAM ORACLE

Definisi Concurrency:

- ✓Akses data oleh beberapa pengguna atau proses secara simultan.
- ✓Kemampuan sistem untuk menangani beberapa transaksi atau permintaan secara simultan.
- ✓Memastikan bahwa banyak pengguna dapat mengakses dan mengubah data secara bersamaan dengan aman.

Masalah Concurrency:

- ✓**Deadlock** (situasi di mana tidak ada transaksi yang dapat melanjutkan, sehingga aktivitas pada database menjadi terhenti dikarenakan dikunci oleh transaksi yang lain)
- ✓**Lost Update** (pembaruan yang dilakukan oleh satu transaksi ditimpa oleh transaksi lain, sehingga perubahan pertama hilang atau tidak tercatat)

MANAJEMEN CONCURRENCY

❖ **Fitur Oracle:**

- ✓ **Locking Mechanism**: seperti *exclusive* dan *shared locks* digunakan untuk mengelola akses bersamaan dan mencegah konflik data selama transaksi.
- ✓ **Undo Tablespace untuk rollback**: menyediakan mekanisme untuk membatalkan transaksi, mendukung konsistensi data, dan memulihkan data saat diperlukan.

❖ **Perintah Penting:**

- ✓ SELECT FOR UPDATE: **mengunci baris data** yang sedang diambil oleh transaksi, sehingga transaksi lain tidak bisa mengubah data tersebut sampai kunci dilepaskan.
- ✓ COMMIT: **menyelesaikan transaksi** dan membuat semua perubahan data yang dilakukan selama transaksi menjadi permanen di database.
- ✓ ROLLBACK: **membatalkan transaksi** dan mengembalikan data ke keadaan sebelum transaksi dimulai.

❖ **Strategi Pengelolaan:**

- ✓ Optimalkan transaksi pendek.
- ✓ Hindari lock yang terlalu lama.

PERCOBAAN 1 MEMBUAT SKEMA BARU

1. Buka Oracle SQL Plus dan hubungkan ke database Anda.
2. Buat skema baru dengan nama **bank_sampah** dan password **secure_pass**.
3. Pastikan skema ini memiliki hak untuk membuat tabel.Berikan kuota sebesar 50 MB pada tablespace USERS.

```
CREATE USER bank_sampah IDENTIFIED BY secure_pass;
GRANT CREATE SESSION, CREATE TABLE TO bank_sampah;
ALTER USER bank_sampah QUOTA 50M ON USERS;
```

PERCOBAAN 2: MEMBUAT OBJEK DALAM SKEMA

1. Hubungkan ke skema **bank_sampah.**
2. Buat tabel **nasabah** dengan struktur berikut:
   - **id_nasabah** (NUMBER) sebagai Primary Key.
   - **nama_nasabah** (VARCHAR2(50)).
   - **saldo** (NUMBER).

```
CREATE TABLE nasabah (
    id_nasabah NUMBER PRIMARY KEY,
    nama_nasabah VARCHAR2(50),
    saldo NUMBER
);
```

3. Tambahkan beberapa data sampel ke dalam tabel **nasabah**.

```
INSERT INTO nasabah (id_nasabah, nama_nasabah, saldo)
VALUES (1, 'Ahmad', 50000),
       (2, 'Budi', 75000),
       (3, 'Citra', 100000);
```

PERCOBAAN 3: MEMBERIKAN HAK AKSES

1. Buat pengguna baru bernama **admin_view** dengan password **view_only**.
2. Berikan **hak akses** kepada pengguna **admin_view** agar dapat melihat tabel nasabah, tetapi tidak dapat mengubah data.

```
CREATE USER admin_view IDENTIFIED BY view_only;
GRANT CREATE SESSION TO admin_view;
GRANT SELECT ON bank_sampah.nasabah TO admin_view;
```

PERCOBAAN 4: MENGUBAH SKEMA

1. Tambahkan kolom baru ke tabel nasabah bernama tanggal_bergabung dengan tipe data DATE.
2. Perbarui data kolom ini dengan nilai tanggal bergabung (gunakan UPDATE).

```
ALTER TABLE nasabah ADD tanggal_bergabung DATE;

UPDATE nasabah
SET tanggal_bergabung = SYSDATE
WHERE id_nasabah = 1;

UPDATE nasabah
SET tanggal_bergabung = TO_DATE('2025-01-01', 'YYYY-MM-DD')
WHERE id_nasabah = 2;

UPDATE nasabah
SET tanggal_bergabung = TO_DATE('2025-02-15', 'YYYY-MM-DD')
WHERE id_nasabah = 3;
```

PERCOBAAN 5: MANAJEMEN CONCURRENCY

Pada tabel nasabah di skema bank_sampah, dua pengguna mencoba memperbarui data saldo untuk id_nasabah = 101 secara bersamaan.Jelaskan bagaimana dapat memastikan bahwa hanya satu transaksi yang dapat memodifikasi data tersebut pada satu waktu.

- Gunakan perintah **SELECT FOR UPDATE** untuk mengunci baris data sehingga transaksi lain harus menunggu hingga kunci dilepaskan.

```
SELECT *
FROM nasabah
WHERE id_nasabah = 101
FOR UPDATE;
```

- Perintah ini memastikan bahwa baris dengan id_nasabah = 101 tidak dapat dimodifikasi oleh transaksi lain hingga transaksi saat ini selesai dengan COMMIT atau ROLLBACK.

PERCOBAAN 6: MENGHINDARI LOST UPDATE

Dua pengguna sedang memperbarui saldo nasabah. Pengguna A menambahkan saldo sebesar 50.000, sementara Pengguna B mengurangi saldo sebesar 30.000 untuk id_nasabah = 102. Jelaskan bagaimana Anda dapat mencegah lost update sehingga kedua pembaruan tetap tercatat dengan benar. Tuliskan perintah SQL.

- Gunakan perintah **SELECT FOR UPDATE** sebelum memperbarui data untuk mencegah konflik antar transaksi

```sql
-- Pengguna A: Menambahkan saldo
SELECT saldo
FROM nasabah
WHERE id_nasabah = 102
FOR UPDATE;

UPDATE nasabah
SET saldo = saldo + 50000
WHERE id_nasabah = 102;

COMMIT;

-- Pengguna B: Mengurangi saldo (setelah Pengguna A selesai)
SELECT saldo
FROM nasabah
WHERE id_nasabah = 102
FOR UPDATE;

UPDATE nasabah
SET saldo = saldo - 30000
WHERE id_nasabah = 102;

COMMIT;
```

KESIMPULAN

Manajemen skema, data, dan concurrency adalah fondasi utama yang memastikan performa database :

- Konsisten: Data tetap akurat dan sesuai integritasnya.
- Efisien: Operasi berjalan cepat tanpa beban kinerja berlebih.
- Skalabel: Database mampu menangani peningkatan jumlah data dan pengguna tanpa mengorbankan performa.Aman: Data dilindungi dari konflik transaksi atau akses yang tidak sah

LATIHAN

KERJAKAN SELURUH PERINTAH SQL PADA LATIHAN DAN CAPTURE PERINTAH SQL BESERTA OUTPUTNYA:

1. Bayangkan bahwa tabel **nasabah** mulai dipenuhi data yang tidak relevan. Anda ingin membuat backup sebelum menghapus beberapa data. Ikuti langkah-langkah berikut:

a. Ekspor tabel **nasabah** ke file dump menggunakan Data Pump.

b. Hapus nasabah yang memiliki saldo kurang dari 60.000.

c. Rollback jika Anda salah menghapus data.

LATIHAN

2. Membuat tabel baru bernama **transaksi** dalam skema **bank_sampah** dengan struktur berikut:
   - **id_transaksi**: NUMBER (Primary Key).
   - **id_nasabah**: NUMBER (Foreign Key, terkait tabel nasabah).
   - **jenis_sampah**: VARCHAR2(50).
   - **berat**: NUMBER.
   - **total_harga**: NUMBER.
   - **tanggal_transaksi**: DATE.

LATIHAN

3. Tambahkan data berikut ke tabel transaksi:
   - ID Transaksi: 1, ID Nasabah: 101, Jenis Sampah: "Plastik", Berat: 5 kg, Total Harga: 25000, Tanggal Transaksi: 2025-03-01.
   - ID Transaksi: 2, ID Nasabah: 102, Jenis Sampah: "Kertas", Berat: 3 kg, Total Harga: 15000, Tanggal Transaksi: 2025-03-02.

LATIHAN

4. Seorang nasabah melaporkan bahwa data berat sampah untuk transaksi ID 1 salah. Berat yang benar adalah 6 kg, sehingga total harga harus diperbarui menjadi 30000. Tuliskan perintah SQL untuk memperbarui data tersebut.
5. Data pada transaksi ID 2 ternyata tidak valid. Tuliskan perintah SQL untuk menghapus data tersebut dari tabel transaksi.
6. Tampilkan semua data transaksi yang memiliki jenis sampah "Plastik" dan berat lebih dari 5 kg. Tuliskan perintah SQL yang digunakan.
7. Anda memasukkan data baru ke tabel nasabah, tetapi terjadi kesalahan sebelum menyimpan transaksi. Jelaskan bagaimana Anda membatalkan perubahan dan tuliskan perintah SQL yang digunakan.

LATIHAN

8. Hitung **total berat sampah** yang disetor dalam tabel **transaksi**. Tuliskan perintah SQL yang digunakan.
9. Tampilkan **nama nasabah** beserta **total berat sampah** yang mereka setor. Asumsikan terdapat
   - tabel **nasabah** dengan kolom **id_nasabah** dan **nama_nasabah**. Tuliskan perintah SQL-nya.
10. Bagaimana Anda memastikan bahwa data dalam tabel transaksi tetap konsisten saat beberapa pengguna mencoba memperbarui data secara bersamaan? Berikan contoh perintah SQL yang digunakan untuk mengunci data