

ORACLE PL/SQL · PERTEMUAN 09 / 14

Dasar PL/SQL

Dari SELECT menjadi blok prosedural dengan variabel, record dan kontrak hasil.

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

Versi PDF materi pertemuan 09. Contoh dan tabel dapat dibaca di PDF. Kuis dapat dijalankan di halaman web: [Buka materi daring](#).

Tujuan pembelajaran

1. Menjelaskan pembagian kerja PL/SQL engine dan SQL engine.
2. Menulis anonymous block DECLARE, BEGIN, EXCEPTION, END dengan delimiter client yang tepat.
3. Menggunakan variabel, konstanta, assignment, %TYPE dan %ROWTYPE.
4. Membaca tepat satu baris melalui SELECT INTO serta membedakan kardinalitas nol dan banyak.
5. Menampilkan pesan diagnostik DBMS_OUTPUT tanpa menganggapnya commit.

Pokok bahasan / praktikum

- Arsitektur PL/SQL engine dan SQL engine
- Anonymous block
- DECLARE BEGIN EXCEPTION END
- Variabel dan konstanta
- Assignment :=
- %TYPE
- %ROWTYPE
- DBMS_OUTPUT
- SELECT INTO dan kardinalitas
- NULL dan scope

Prasyarat: Selesaikan modul 02/03 atau jalankan setup bersama sekali pada schema latihan sendiri. Gunakan Oracle XE lokal melalui SQL Developer ATAU Oracle FreeSQL setelah login pada schema pribadi yang diizinkan; tidak harus memakai keduanya.

Jangan menjalankan ulang CREATE TABLE/seed pada objek yang sudah ada.
Selesaikan modul SQL sebelumnya; modul 10 juga memerlukan dasar blok modul 09.

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. Apa yang dilakukan engine ketika sebuah blok berisi assignment dan SELECT?
2. Mengapa sebuah SELECT biasa dapat menampilkan banyak baris tetapi SELECT INTO satu variabel tidak?
3. Apakah pesan berhasil berarti data sudah tersimpan permanen?

Uji pemahaman awal

1. Operator assignment variabel PL/SQL adalah ...

- ☐ =
- ☐ ==
- ☐ :=
- ☐ =>

2. SELECT INTO biasa tanpa baris memicu ...

- NO_DATA_FOUND
- TOO_MANY_ROWS
- COMMIT otomatis
- Nilai selalu nol

3. Apa fungsi produk_demo%ROWTYPE?

- Membuat tabel baru
- Menghapus constraint
- Menjalankan SELECT otomatis
- Mendefinisikan record dengan field satu baris tabel

PAHAMI KONSEP · BACA TERSTRUKTUR · TERAPKAN

Teori dan pembahasan

PL/SQL menambahkan variabel, alur prosedural dan penanganan error di atas SQL. Mulailah dari anonymous block yang hanya membaca data.

Konsep inti bab ini

- DECLARE dan EXCEPTION opsional; bagian BEGIN–END wajib.
- Assignment memakai :=; perbandingan memakai =.
- SELECT INTO biasa memerlukan tepat satu baris.

Urutan pembahasan

- 01 Arsitektur engine dan batas SQL
- 02 Struktur anonymous block dan client
- 03 Variabel, konstanta dan assignment
- 04 Anchored type dan record
- 05 SELECT INTO dan kardinalitas
- 06 DBMS_OUTPUT, scope dan pembuktian

Gambaran konsep dalam satu tabel

Bagian / fitur	Fungsi	Perhatian
DECLARE	Variabel, konstanta dan record	Nilai awal dan scope
BEGIN – END	Pernyataan yang dijalankan	Akhiri statement dengan titik koma
EXCEPTION	Menangani kegagalan	Tidak kembali ke statement yang gagal
%TYPE / %ROWTYPE	Mengikuti tipe kolom/baris	Tidak menyalin semua aturan bisnis
DBMS_OUTPUT	Pesan ke buffer client	Bukan bukti COMMIT

01 Arsitektur engine dan batas SQL

PL/SQL memperluas SQL dengan deklarasi, langkah prosedural dan penanganan exception.

Pada latihan database ini, client mengirim blok ke Oracle Database. PL/SQL engine menjalankan bagian prosedural seperti assignment, sedangkan pernyataan SQL dikerjakan oleh SQL engine.

POIN PEMBAHASAN

- Keduanya bekerja bersama; sebuah variabel tidak menjadi kolom tabel dan `DBMS_OUTPUT` bukan hasil `SELECT` biasa.
- Perpindahan antara kerja prosedural dan SQL mempunyai biaya, sehingga jangan mengubah operasi himpunan sederhana menjadi pengulangan per baris tanpa alasan.
- Anonymous block berguna untuk percobaan karena tidak disimpan sebagai objek schema bernama.
- Menjalankan blok berikut tidak memerlukan `CREATE PROCEDURE`; cukup izin sesi dan akses tabel latihan milik sendiri.

02 Struktur anonymous block dan client

DECLARE membuka bagian deklarasi yang opsional.

BEGIN membuka bagian eksekusi yang wajib, **EXCEPTION** membuka handler opsional, dan **END** diakhiri titik koma. Setiap assignment atau pemanggilan prosedur juga diakhiri titik koma.

POIN PEMBAHASAN

- Slash pada baris tersendiri merupakan delimiter client Run Script, bukan operator akhir bahasa PL/SQL.
- Dalam SQL Developer gunakan Run Script untuk skrip dengan slash, atau jalankan blok sebagai satu statement sesuai mode editor.
- Pada FreeSQL yang sudah login, ikuti pemisahan blok dan dukungan output editornya; jangan mengirim slash sebagai bagian isi blok jika runner memisahkannya sendiri.
- Contoh sengaja hanya membaca tabel sehingga tidak membutuhkan keputusan commit.

03 Variabel, konstanta dan assignment

Deklarasikan nama variabel bersama tipe, ukuran jika perlu, dan nilai awal.

Variabel tanpa nilai awal biasanya `NULL`, sehingga operasi aritmetika terhadapnya dapat tetap `NULL`. Assignment memakai `:=`, sedangkan `=` digunakan untuk perbandingan.

POIN PEMBAHASAN

- Konstanta diberi kata `CONSTANT` dan nilai awal; nilainya tidak boleh diubah kemudian.
- Contoh menghitung nilai dasar dan diskon sebagai ekspresi sederhana, bukan kebijakan uang produksi.
- Gunakan tipe `NUMBER` berskala untuk nilai uang dan validasi input positif dengan maksimal dua desimal sebelum penyimpanan ketika menerima parameter.
- Literal teks memakai tanda petik tunggal.
- Oracle memperlakukan string kosong sebagai `NULL`; gunakan `IS NULL` jika maksudnya memeriksa ketiadaan nilai, bukan membandingkan `= ''`.

04 Anchored type dan record

`%TYPE` mengikuti tipe kolom atau variabel yang dirujuk, sehingga perubahan tipe kolom tidak mengharuskan penulisan ulang tipe variabel secara manual. `v_nama_pelanggan_demo.nama %TYPE` cocok untuk nama pelanggan.

`%ROWTYPE` membuat record dengan field sesuai satu baris tabel; `r_produk_produk_demo %ROWTYPE` menyediakan `id_produk`, `nama_produk`, `harga` dan `stok`. Akses field dengan titik, misalnya `r_produk.harga`.

POIN PEMBAHASAN

- Anchoring tipe bukan menyalin seluruh aturan bisnis: constraint atau default kolom tidak otomatis menjadi aturan variabel record.
- `SELECT * INTO` record tepat untuk latihan satu baris dengan bentuk tabel yang sesuai; laporan biasa tetap sebaiknya memilih kolom eksplisit.
- Gunakan nama berawalan `v_` atau `r_` agar tidak bertabrakan dengan kolom dalam `WHERE`.

05 SELECT INTO dan kardinalitas

SELECT INTO biasa harus menghasilkan tepat satu baris, dengan jumlah dan tipe kolom yang sesuai target.

Pencarian primary key produk 11 memenuhi kontrak pada seed. Nol baris memicu **NO_DATA_FOUND**, sedangkan banyak baris memicu **TOO_MANY_ROWS**; jangan menganggap hasil pertama dipilih otomatis.

POIN PEMBAHASAN

- Aggregate **COUNT (*)** tanpa **GROUP BY** menghasilkan satu baris bahkan bila tidak ada data cocok, tetapi **AVG** atau **SUM** pada himpunan kosong dapat bernilai **NULL**.
- Menambahkan **GROUP BY** mengubah kembali jumlah baris sehingga harus dianalisis.
- Contoh memakai subblok agar setiap percobaan kardinalitas dapat memiliki handler sendiri.
- Setelah handler berjalan, eksekusi tidak kembali ke pernyataan yang gagal; ia meninggalkan blok yang menangani exception.

06 DBMS_OUTPUT, scope dan pembuktian

DBMS_OUTPUT.PUT_LINE menaruh teks ke buffer yang perlu dibaca client.

Aktifkan panel DBMS Output untuk koneksi SQL Developer, atau gunakan **SET SERVEROUTPUT ON** ketika client script mendukungnya. Pada FreeSQL aktifkan tampilan output sesuai editor; **SET SERVEROUTPUT** adalah perintah client, bukan SQL database universal.

POIN PEMBAHASAN

- Pesan biasanya tampil setelah blok selesai, bukan sebagai streaming kemajuan transaksi.
- Variabel berlaku dalam scope blok tempat dideklarasikan dan tidak tersedia pada blok berikutnya.

- Catat SQL serta output aktual saat mencoba; contoh yang disediakan belum merupakan bukti eksekusi pada akun Oracle mahasiswa.
- Pesan berhasil juga tidak membuktikan DML telah commit.
- Pemilik transaksi tetap harus memilih `COMMIT` atau `ROLLBACK` ketika latihan selanjutnya mulai mengubah data.

Alur mencoba anonymous block

- 1 Deklarasikan data
- 2 Baca dengan `SELECT INTO`
- 3 Lakukan assignment
- 4 Tampilkan `DBMS_OUTPUT`
- 5 Tangani kondisi yang diharapkan

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.

SELECT INTO: satu baris, nol baris dan banyak baris

Gunakan produk seed untuk memahami kontrak `SELECT INTO` sebelum memproses datanya.

Contoh SQL / PL/SQL

```
DECLARE
  v_nama produk_demo.nama_produk%TYPE;
BEGIN
  SELECT nama_produk INTO v_nama
  FROM produk_demo WHERE id_produk = 11;
  DBMS_OUTPUT.PUT_LINE(v_nama);
END;
/
```

Kondisi query	Jumlah baris seed	Akibat pada SELECT INTO
id_produk = 11	1	Variabel menerima Kopi
id_produk = 999	0	NO_DATA_FOUND
harga > 10000	3	TOO_MANY_ROWS

Cara membaca contoh

1. `DECLARE` menyiapkan `v_nama` dengan tipe kolom `nama_produk`.
2. `SELECT ... INTO` mengisi variabel jika tepat satu baris ditemukan.
3. `DBMS_OUTPUT` hanya menampilkan pesan; output yang terlihat bukan `COMMIT`.

Yang perlu dipahami: Query banyak baris membutuhkan cara pembacaan yang sesuai, misalnya cursor pada pertemuan11.

Video pembelajaran

Video utama — fokus bab ini

Struktur anonymous block `DECLARE–BEGIN–EXCEPTION–END` dan `DBMS_OUTPUT`. Referensi tambahan memperluas variabel, konstanta, `SELECT` dan `%TYPE/%ROWTYPE`.

PL/SQL Block Structure | Anonymous Block | How to write Hello World Program in PL/SQL| TechnonTechTV

TechnonTechTV · [Buka video utama di YouTube](#)

Video tambahan sesuai subtopik

1. [MASTER Oracle PLSQL in 30 Days or LESS!](#)

NIC IT ACADEMY

Fokus: Variabel, konstanta, SELECT/DML dalam blok dan %TYPE/%ROWTYPE.

Rentang yang disarankan: 00:29:08–01:42:38; lanjut control structure pada pertemuan10.

Batas cakupan dan panduan menonton: Video utama memakai Oracle11g untuk konsep blok. Pilih bagian fundamental pada referensi panjang, bukan seluruh kursus; langkah editor menyesuaikan XE/FreeSQL.

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.

Gunakan tabel pertemuan02 dan data pertemuan03 atau setup index sekali. Periksa keadaan data bila latihan sebelumnya pernah di-commit. Jangan menjalankan ulang CREATE TABLE/INSERT atau menghapus data secara buta.

Langkah 1 — Anonymous block paling sederhana

```
BEGIN
  DBMS_OUTPUT.PUT_LINE('Blok PL/SQL berjalan');
END;
/
```

Aktifkan output editor. Slash untuk Run Script, bukan bagian bahasa PL/SQL. Tidak membaca atau mengubah tabel.

Langkah 2 — DECLARE, variabel, konstanta dan assignment

```
DECLARE
  v_jumlah PLS_INTEGER := 2;
  v_harga NUMBER(12,2) := 30000;
  c_diskon CONSTANT NUMBER := 0.10;
  v_total NUMBER(12,2);
BEGIN
  v_total := v_jumlah * v_harga;
  v_total := v_total * (1 - c_diskon);
  DBMS_OUTPUT.PUT_LINE('Jumlah = ' || v_jumlah);
  DBMS_OUTPUT.PUT_LINE('Total setelah diskon = ' || v_total);
END;
/
```

c_diskon tetap, sedangkan v_total berubah melalui :=. Format angka output mengikuti pengaturan sesi; verifikasi nilai, bukan bentuk teks lokal tertentu.

Langkah 3 — %TYPE dan SELECT INTO satu pelanggan

```
DECLARE
  v_id pelanggan_demo.id_pelanggan%TYPE := 1;
  v_nama pelanggan_demo.nama%TYPE;
```

```

v_kota pelanggan_demo.kota%TYPE;
BEGIN
    SELECT nama, kota INTO v_nama, v_kota
    FROM pelanggan_demo WHERE id_pelanggan = v_id;
    DBMS_OUTPUT.PUT_LINE(v_nama || ' / ' || NVL(v_kota, 'Kota belum
diisi'));
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('ID pelanggan tidak ditemukan');
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Query melanggar kontrak satu baris');
END;
/

```

Pada seed membaca Ayu. PK menjamin maksimal satu baris; handler tetap menjelaskan kontrak jika query diubah.

Langkah 4 — %ROWTYPE untuk satu produk

```

DECLARE
    r_produk produk_demo%ROWTYPE;
BEGIN
    SELECT * INTO r_produk FROM produk_demo WHERE id_produk = 11;
    DBMS_OUTPUT.PUT_LINE(r_produk.nama_produk || ' stok=' ||
r_produk.stok);
    DBMS_OUTPUT.PUT_LINE('Harga=' || r_produk.harga);
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Produk tidak ditemukan');
END;
/

```

Record menampung satu baris produk dan field diakses memakai titik. Constraint tabel tidak otomatis memvalidasi assignment ke field record.

Langkah 5 — DECLARE BEGIN EXCEPTION END: nol dan banyak baris

```

DECLARE
    v_nama pelanggan_demo.nama%TYPE;
BEGIN
    BEGIN
        SELECT nama INTO v_nama FROM pelanggan_demo WHERE 1 = 0;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Percobaan nol baris: NO_DATA_FOUND');
    END;
    BEGIN
        SELECT nama INTO v_nama FROM pelanggan_demo;
        DBMS_OUTPUT.PUT_LINE('Tabel ternyata hanya berisi satu
baris');
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Tabel kosong');
        WHEN TOO_MANY_ROWS THEN
            DBMS_OUTPUT.PUT_LINE('Percobaan banyak baris:
TOO_MANY_ROWS');
    END;
    DBMS_OUTPUT.PUT_LINE('Blok luar tetap dilanjutkan');
END;
/

```

WHERE 1=0 selalu menunjukkan kasus nol; seed pelanggan menunjukkan kasus banyak. Subblok terpisah memungkinkan percobaan berikutnya berjalan.

Langkah 6 — Aggregate satu baris dan NULL string kosong

```

DECLARE
    v_banyak NUMBER;
    v_teks VARCHAR2(20) := '';
BEGIN
    SELECT COUNT(*) INTO v_banyak FROM penjualan_demo WHERE 1 = 0;
    DBMS_OUTPUT.PUT_LINE('COUNT kosong = ' || v_banyak);
    DBMS_OUTPUT.PUT_LINE('Teks kosong ditampilkan sebagai: ' ||
    NVL(v_teks, 'NULL'));

```

```
END;  
/
```

COUNT tanpa GROUP BY tetap satu baris bernilai nol. Oracle menganggap string kosong sebagai NULL; NVL hanya memberi label tampilan.

Tugas / latihan

1. Pada XE ATAU FreeSQL schema pribadi, buat anonymous block dengan variabel, konstanta dan assignment; lampirkan output aktual dan cara mengaktifkan output.
2. Gunakan %TYPE untuk mengambil satu pelanggan dan %ROWTYPE untuk satu produk; jelaskan syarat jumlah baris.
3. Uji SELECT INTO untuk tepat satu, nol dan banyak baris dalam subblok berhandler. Bandingkan dengan COUNT(*) tanpa GROUP BY pada himpunan kosong.

Pertanyaan analisis

1. Mengapa menyalin %TYPE tidak berarti semua aturan CHECK dan DEFAULT tabel telah menjadi aturan variabel?
2. Jelaskan jalur client, PL/SQL engine, SQL engine dan buffer DBMS_OUTPUT pada satu blok SELECT INTO.

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 19c — Architecture of PL/SQL](#)
- [Oracle 19c — Language Fundamentals](#)
- [Oracle 19c — SELECT INTO](#)
- [Oracle 19c — DBMS_OUTPUT](#)

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