

ORACLE PL/SQL · PERTEMUAN 05 / 14

Function, Aggregation & Grouping

Mengolah teks, angka, tanggal, dan ringkasan tanpa mengandalkan konversi tersembunyi.

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

Versi PDF materi pertemuan 05. Contoh dan tabel dapat dibaca di PDF. Kuis dapat dijalankan di halaman web: [Buka materi daring](#).

Tujuan pembelajaran

1. Menerapkan fungsi huruf, substring, panjang, pembulatan dan pemotongan.
2. Mengolah tanggal dengan SYSDATE, ADD_MONTHS, dan MONTHS_BETWEEN.
3. Menggunakan TO_CHAR, TO_DATE, dan TO_NUMBER secara eksplisit dengan kewaspadaan NLS.
4. Menghitung COUNT, SUM, AVG, MIN, MAX dan membedakan WHERE dari HAVING.

Pokok bahasan / praktikum

- UPPER LOWER INITCAP SUBSTR LENGTH
- ROUND TRUNC angka
- SYSDATE ADD_MONTHS MONTHS_BETWEEN
- TO_CHAR TO_DATE TO_NUMBER
- Format eksplisit dan NLS
- COUNT SUM AVG MIN MAX
- GROUP BY
- HAVING
- NULL pada agregasi

Prasyarat: Modul02/03 atau setup indeks sekali serta modul04. Jalankan di XE lokal ATAU FreeSQL login; query modul ini tidak mengubah seed.

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. Apakah jumlah pelanggan sama dengan jumlah email yang terisi?
2. Mengapa teks 1.500 bisa ditafsirkan berbeda pada dua komputer?
3. Bagaimana menyaring kelompok penjualan, bukan baris penjualannya?

Uji pemahaman awal

1. Berapa COUNT(email) pada seed pelanggan?

- ☐ 0
- ☐ 2
- ☐ 3
- ☐ 4

2. Apa beda WHERE dan HAVING?

- ☐ Keduanya hanya menyaring kolom
- ☐ WHERE setelah agregasi
- ☐ WHERE menyaring baris, HAVING menyaring kelompok
- ☐ HAVING mengganti ORDER BY

3. Konversi input tanggal yang paling jelas adalah?

- Mengandalkan format tanggal komputer
- TO_DATE dengan model format yang sesuai
- Mengutip angka lalu berharap konversi
- TO_CHAR selalu menghasilkan DATE

PAHAMI KONSEP · BACA TERSTRUKTUR · TERAPKAN

Teori dan pembahasan

Mulai dari transformasi nilai per baris, kemudian beralih ke ringkasan kelompok. Jenis fungsi dan penempatan filter menentukan arti hasil laporan.

Konsep inti bab ini

- Single-row function dan aggregate memiliki lingkup kerja berbeda.
- COUNT (*) dan COUNT (kolom) tidak selalu memberikan jumlah yang sama.
- Konversi tanggal/angka perlu format eksplisit agar tidak bergantung asumsi NLS.

Urutan pembahasan

- 01** Fungsi per baris dan teks
- 02** ROUND dan TRUNC untuk angka
- 03** Tanggal dan selisih bulan
- 04** Konversi eksplisit dan pengaruh NLS
- 05** Agregasi dan NULL
- 06** GROUP BY, WHERE, dan HAVING

Gambaran konsep dalam satu tabel

Kelompok	Contoh	Lingkup hasil
Teks	UPPER, LOWER, INITCAP, SUBSTR, LENGTH	Satu nilai per baris
Angka / tanggal	ROUND, TRUNC, SYSDATE, ADD_MONTHS	Transformasi nilai
Konversi	TO_CHAR, TO_DATE, TO_NUMBER	Perubahan tipe dengan format
Agregasi	COUNT, SUM, AVG, MIN, MAX	Ringkasan kumpulan/kelompok
Filter laporan	WHERE / HAVING	Sebelum / sesudah agregasi

01 Fungsi per baris dan teks

Fungsi skalar mengubah nilai satu baris menjadi nilai hasil, sedangkan agregasi meringkas sekumpulan baris.

UPPER mengubah teks menjadi huruf besar, LOWER menjadi huruf kecil, dan INITCAP membuat awal kata kapital sesuai aturan fungsi. Penggunaan fungsi pada `SELECT` tidak mengganti nama yang tersimpan. `UPPER(nama) LIKE 'A%'` dapat menormalkan pencarian sederhana, tetapi aturan bahasa dan collation tetap perlu diperhatikan untuk teks yang lebih kompleks.

POIN PEMBAHASAN

- `SUBSTR(nama,1,3)` mengambil paling banyak tiga karakter mulai posisi satu; posisi negatif dapat dihitung dari akhir teks.
- `LENGTH` menghitung panjang karakter, bukan selalu jumlah byte penyimpanan.
- Pada seed, `LENGTH('Ayu')` adalah tiga dan `SUBSTR('Bandung',1,3)` adalah Ban.
- Fungsi teks pada `NULL` umumnya menghasilkan `NULL`; tampilan kosong tidak berarti string berpanjang nol tersimpan secara mandiri pada Oracle.

02 ROUND dan TRUNC untuk angka

ROUND(angka,2) membulatkan ke dua tempat desimal, sedangkan **TRUNC(angka,2)** memotong pecahan setelah dua tempat tanpa pembulatan.

Contoh 123.456 menghasilkan 123.46 atau 123.45. Argumen negatif mengolah posisi di sebelah kiri tanda desimal; **ROUND(12345,-2)** membulatkan ratusan. Fungsi numerik tidak sama dengan format tampilan: nilai 30000 dapat ditampilkan tanpa dua nol pecahan meskipun kolom harga memiliki scale dua.

POIN PEMBAHASAN

- Gunakan **TO_CHAR** jika laporan memerlukan tampilan tetap, bukan mengubah semua nilai menjadi teks sebelum menghitung.
- Pembulatan total dan pembulatan setiap komponen bisa memberikan perbedaan; jelaskan tingkat pembulatan yang dimaksud.
- Harga pada dataset tetap **NUMBER**, dan hasil fungsi tidak mengubah baris kecuali sengaja dipakai pada **UPDATE** yang dibahas sebelumnya.
- Modul ini seluruhnya bersifat baca.

03 Tanggal dan selisih bulan

SYSDATE mengembalikan tanggal dan waktu server database, bukan selalu zona waktu perangkat mahasiswa.

Hasilnya berubah sesuai waktu eksekusi, sehingga jangan menyalin tanggal hasil sebagai jawaban tetap. **ADD_MONTHS** menambah atau mengurangi bulan kalender; penanganan akhir bulan mempertimbangkan panjang bulan tujuan. Menambah satu bulan ke 31 Januari dapat menghasilkan hari terakhir Februari, bukan mempertahankan angka hari yang tidak tersedia.

POIN PEMBAHASAN

- **MONTHS_BETWEEN** mengembalikan selisih bulan yang dapat berupa pecahan.
- Jika kedua tanggal mempunyai hari yang sama atau keduanya akhir bulan, hasilnya bilangan bulan bulat; kasus lain memakai aturan pecahan fungsi, bukan sekadar jumlah hari dibagi tiga puluh.

- Contoh 1 Desember dan 1 Oktober memberi dua.
- Beberapa aturan kalender dipengaruhi NLS_CALENDAR, jadi contoh mengasumsikan kalender Gregorian; catat jika sesi Anda menggunakan kalender berbeda.

04 Konversi eksplisit dan pengaruh NLS

TO_CHAR mengubah angka atau tanggal menjadi teks dengan model format.

TO_DATE mengubah teks menjadi DATE, bukan mengubah DATE menjadi DATE.

TO_NUMBER membaca teks sebagai angka. Gunakan **TO_DATE** ('2026-10-01','FXYYYY-MM-DD') untuk input yang diharapkan cocok persis, atau literal DATE untuk konstanta SQL. Format tanggal sesi dapat berubah; jangan mengandalkan '01/10/2026' tanpa model yang menyatakan urutan hari dan bulan.

POIN PEMBAHASAN

- NLS_DATE_FORMAT mengatur format tanggal default, NLS_DATE_LANGUAGE memengaruhi nama bulan, dan NLS_NUMERIC_CHARACTERS menentukan pemisah desimal serta kelompok.
- Contoh konversi angka menyebut model dan parameter NLS secara eksplisit agar 1,500.25 bermakna 1500.25.
- Model FM menghilangkan padding pada tampilan angka.
- Jangan memakai teks berformat mata uang sebagai nilai perhitungan.
- Konversi tidak valid menimbulkan error, bukan otomatis **NULL** atau nol yang aman.

05 Agregasi dan NULL

COUNT (*) menghitung baris; **COUNT (email)** menghitung hanya email non-**NULL**.

Karena Citra belum memiliki email, seed pelanggan memberi tiga baris tetapi dua email. **SUM** menjumlahkan, **AVG** menghitung rata-rata nilai non-**NULL**, **MIN** mencari terkecil, dan **MAX** terbesar. Pada empat produk, harga berjumlah 67000 dan rata-ratanya 16750. Angka ini adalah prediksi dari seed, bukan klaim bahwa Oracle telah dijalankan penyusun.

POIN PEMBAHASAN

- `NULL` biasanya diabaikan oleh agregasi, bukan diperlakukan sebagai nol.
- Pada kumpulan tanpa baris, `COUNT` menghasilkan nol sedangkan `SUM`, `AVG`, `MIN`, dan `MAX` menghasilkan `NULL`.
- `SUM` (jumlah * harga) pada detail menjumlahkan nilai transaksi menjadi 510000 pada seed.
- `COUNT (DISTINCT id_pelanggan)` pada penjualan menghitung dua pelanggan teridentifikasi; penjualan anonim tetap dihitung `COUNT (*)` tetapi tidak menambah hitungan pelanggan non-`NULL`.

06 GROUP BY, WHERE, dan HAVING

GROUP BY membentuk kelompok menurut kolom atau ekspresi.

Setiap item `SELECT` yang bukan agregat harus sesuai pengelompokan, sehingga memilih nama produk bebas bersama `SUM` tanpa kelompok yang benar dapat memicu ORA-00979. `GROUP BY` status pada penjualan menghasilkan BARU dengan dua baris dan LUNAS dengan satu. `NULL` pada kolom kelompok dikumpulkan sebagai satu kelompok meskipun `NULL` bukan nilai yang dapat dibandingkan dengan tanda sama dengan.

POIN PEMBAHASAN

- `WHERE` menyaring baris sebelum agregasi, sedangkan `HAVING` menyaring hasil kelompok setelah agregasi.
- `HAVING COUNT (*) >= 2` mempertahankan kelompok BARU pada seed.
- Alias agregat di `SELECT` jangan digunakan sebagai pengganti ekspresi dalam `HAVING` untuk kompatibilitas 19c.
- `ORDER BY` boleh memakai alias untuk mengurutkan laporan.
- Kelompok dan total tidak otomatis terurut; nyatakan urutan secara eksplisit.
- Tetap bedakan pertanyaan jumlah transaksi, jumlah pelanggan, dan jumlah unit terjual.

Alur membangun laporan kelompok

- 1 Pilih baris dengan `WHERE`
- 2 Bentuk `GROUP BY`
- 3 Hitung aggregate
- 4 Filter kelompok dengan `HAVING`
- 5 Urutkan hasil laporan

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.

Laporan jumlah dan nilai penjualan per pelanggan

Citra tidak mempunyai penjualan pada seed. Ringkasan tetap menampilkan Citra dengan LEFT JOIN.

Contoh SQL / PL/SQL

```
WITH total_jual AS (  
  SELECT id_jual, SUM(jumlah * harga) AS total  
  FROM detail_jual_demo GROUP BY id_jual  
)  
SELECT p.nama, COUNT(j.id_jual) AS jumlah_penjualan,  
       NVL(SUM(t.total),0) AS total  
FROM pelanggan_demo p  
LEFT JOIN penjualan_demo j ON j.id_pelanggan = p.id_pelanggan  
LEFT JOIN total_jual t ON t.id_jual = j.id_jual
```

```
GROUP BY p.id_pelanggan, p.nama
ORDER BY p.id_pelanggan;
```

Pelanggan	Jumlah penjualan	Total
Ayu	1	300000
Bima	1	150000
Citra	0	0

Cara membaca contoh

1. **GROUP BY** membentuk satu kelompok untuk setiap pelanggan.
2. **COUNT (j.id_jual)** menghitung transaksi yang benar-benar ditemukan, bukan **COUNT (*)**.
3. **SUM** tanpa transaksi menghasilkan **NULL**; **NVL** pada query Oracle mengubah tampilan total menjadi 0.

Yang perlu dipahami: Penjualan 103 tidak memiliki pelanggan; nilainya tidak masuk ke kelompok Ayu, Bima atau Citra. Nilai total berasal dari detail pada query Oracle berikut.

Video pembelajaran

Video utama — fokus bab ini

Aggregate, GROUP BY dan HAVING di Oracle; video tambahan melengkapi single-row function serta konversi data.

Aggregate Functions, Group By and Having in Oracle SQL

Explore IT · [Buka video utama di YouTube](#)

Video tambahan sesuai subtopik

1. [Single Row Functions in SQL | Oracle SQL fundamentals](#)
DBA Genesis

Fokus: Pengenalan kategori character, numeric, date dan conversion single-row functions Oracle.

2. [Oracle Tutorial - Conversion Functions TO_DATE | TO_CHAR | TO_NUMBER](#)

Tech Acad

Fokus: Contoh konversi Oracle TO_DATE, TO_CHAR dan TO_NUMBER.

Batas cakupan dan panduan menonton: Pengenalan single-row function tidak menggantikan seluruh contoh fungsi string, angka dan tanggal; ikuti format NLS eksplisit pada praktikum.

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 — Teks, ROUND dan TRUNC

```
SELECT nama, UPPER(nama) AS huruf_besar, LOWER(nama) AS
huruf_kecil,
  INITCAP(LOWER(nama)) AS awal_kapital, SUBSTR(nama,1,3) AS
tiga_awal,
  LENGTH(nama) AS panjang FROM pelanggan_demo ORDER BY
id_pelanggan;
SELECT ROUND(123.456,2) AS pembulatan, TRUNC(123.456,2) AS
pemotongan,
  ROUND(12345,-2) AS ratusan FROM dual;
SELECT nama_produk, harga, ROUND(harga / 7,2) AS dibulatkan,
  TRUNC(harga / 7,2) AS dipotong FROM produk_demo ORDER BY
id_produk;
```

Query menunjukkan perbedaan operasi numerik, bukan mengubah harga. Untuk nilai 123.456 hasil diprediksi 123.46 dan 123.45.

Langkah 2 — Tanggal dan konversi dengan format eksplisit

```
SELECT TO_CHAR(SYSDATE, 'YYYY-MM-DD HH24:MI:SS') AS waktu_server,
  TO_CHAR(ADD_MONTHS(DATE '2026-01-31',1), 'YYYY-MM-DD') AS
bulan_berikut,
  MONTHS_BETWEEN(DATE '2026-12-01', DATE '2026-10-01') AS beda_bulan
FROM dual;
SELECT TO_DATE('2026-10-01', 'FXYYYY-MM-DD') AS tanggal_input,
  TO_CHAR(DATE '2026-10-01', 'DD MON
YYYY', 'NLS_DATE_LANGUAGE=ENGLISH') AS label_tanggal,
  TO_NUMBER('1,500.25', '9G999D99', 'NLS_NUMERIC_CHARACTERS='''.'''')
AS angka_input,
  TO_CHAR(1500.25, 'FM9G999D00', 'NLS_NUMERIC_CHARACTERS='''.'''') AS
label_angka FROM dual;
SELECT parameter, value FROM nls_session_parameters
WHERE parameter IN
('NLS_DATE_FORMAT', 'NLS_DATE_LANGUAGE', 'NLS_NUMERIC_CHARACTERS', 'N
LS_CALENDAR')
ORDER BY parameter;
```

Pada kalender Gregorian, ADD_MONTHS menghasilkan 2026-02-28 dan beda_bulan 2. Tampilan tanggal_input tetap mengikuti format editor; nilainya DATE. Tidak ada ALTER SESSION yang mengubah kebijakan lingkungan.

Langkah 3 — Ringkasan, kelompok dan HAVING

```
SELECT COUNT(*) AS semua_pelanggan, COUNT(email) AS email_terisi
FROM pelanggan_demo;
SELECT COUNT(*) AS banyak_produk, SUM(harga) AS jumlah_harga,
AVG(harga) AS rata_harga,
  MIN(harga) AS termurah, MAX(harga) AS termahal FROM produk_demo;
SELECT SUM(jumlah * harga) AS nilai_penjualan FROM
detail_jual_demo;
SELECT COUNT(*) AS transaksi, COUNT(DISTINCT id_pelanggan) AS
pelanggan_teridentifikasi FROM penjualan_demo;
SELECT status, COUNT(*) AS banyak FROM penjualan_demo
WHERE tanggal >= DATE '2026-10-01'
GROUP BY status HAVING COUNT(*) >= 2 ORDER BY banyak DESC, status;
SELECT kota, COUNT(*) AS banyak FROM pelanggan_demo GROUP BY kota
ORDER BY kota NULLS LAST;
SELECT COUNT(*) AS banyak, SUM(harga) AS total FROM produk_demo
WHERE id_produk = -1;
```

Diprediksi pelanggan 3/email 2, total harga 67000, rata 16750, min 10000/max 30000, nilai penjualan 510000. Baris terakhir memperlihatkan COUNT nol versus SUM NULL pada sumber kosong.

Tugas / latihan

1. Pada XE lokal ATAU FreeSQL login, jalankan fungsi teks dan numerik; jelaskan perbedaan ROUND/TRUNC dan NULL pada LENGTH.
2. Konversikan satu tanggal dan satu angka memakai format eksplisit, dokumentasikan parameter NLS sesi tanpa mengubahnya, serta jelaskan hasil SYSDATE sebagai waktu server.
3. Buat laporan kelompok status dengan COUNT dan HAVING, laporan nilai detail dengan SUM, serta ringkasan harga dengan AVG/MIN/MAX; bandingkan COUNT(*) dan COUNT(email).

Pertanyaan analisis

1. Mengapa rata-rata nilai yang belum diketahui tidak otomatis sama dengan rata-rata yang memakai nol?
2. Apakah jumlah pelanggan unik pada penjualan sama dengan jumlah transaksi?
Gunakan penjualan anonim untuk menjelaskan.

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: Functions](#)
- [Oracle Database 19c SQL Language Reference: Single-Row-Functions](#)
- [Oracle Database 19c SQL Language Reference: Aggregate-Functions](#)
- [Oracle Database 19c SQL Language Reference: ADD_MONTHS](#)
- [Oracle Database 19c SQL Language Reference: MONTHS_BETWEEN](#)
- [Oracle Database 19c SQL Language Reference: TO_CHAR-datetime](#)
- [Oracle Database 19c SQL Language Reference: TO_DATE](#)

- [Oracle Database 19c SQL Language Reference: TO_NUMBER](#)
- [Oracle Database 19c SQL Language Reference: SELECT](#)

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