

ORACLE PL/SQL · PERTEMUAN 07 / 14

Subquery & Query Lanjutan

Menyusun pertanyaan berlapis, himpunan hasil, dan laporan dengan WITH.

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

Versi PDF materi pertemuan 07. Contoh dan tabel dapat dibaca di PDF. Kuis dapat dijalankan di halaman web: [Buka materi daring](#).

Tujuan pembelajaran

1. Membedakan single-row, multi-row, nested dan correlated subquery.
2. Memilih IN, EXISTS, ANY dan ALL serta mengantisipasi NULL pada NOT IN.
3. Membandingkan UNION, UNION ALL, INTERSECT dan MINUS.
4. Memecah agregasi laporan menjadi CTE WITH yang mudah diuji.

Pokok bahasan / praktikum

- Single-row subquery
- Multi-row subquery
- Nested subquery
- Correlated subquery
- IN
- EXISTS
- ANY
- ALL
- NULL dan NOT IN
- UNION
- UNION ALL
- INTERSECT
- MINUS
- CTE WITH

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. Pahami SELECT, WHERE, NULL dan agregasi.

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 menemukan produk yang lebih mahal daripada harga rata-rata?
2. Mengapa pelanggan tanpa transaksi justru dapat hilang dari NOT IN?
3. Kapan dua hasil SELECT sebaiknya ditumpuk, bukan digabung melalui JOIN?

Uji pemahaman awal

1. Operator yang mempertahankan duplikasi saat menumpuk hasil adalah ...

- UNION
- INTERSECT
- UNION ALL
- MINUS

2. Perbaiki aman untuk mencari pelanggan tanpa penjualan ketika ada pelanggan transaksi NULL adalah ...

- NOT EXISTS dengan korelasi ID
- Mengganti NULL menjadi string kosong
- Menambahkan ORDER BY saja
- Mengganti NOT IN dengan =

3. Apa arti harga > ANY dari kumpulan harga non-NULL yang tidak kosong?

- Lebih besar daripada semua harga
- Lebih besar daripada setidaknya satu harga
- Sama dengan harga rata-rata
- Selalu TRUE

PAHAMI KONSEP · BACA TERSTRUKTUR · TERAPKAN

Teori dan pembahasan

Pecah pertanyaan menjadi hasil antara yang dapat diuji. Kontrak jumlah baris, nullability dan operator himpunan menentukan pilihan bentuk query.

Konsep inti bab ini

- Scalar/single-row subquery tidak boleh menghasilkan banyak baris.
- NOT IN perlu perhatian ketika subquery mengandung NULL.
- WITH menamai query dalam satu statement; tidak otomatis membuat tabel.

Urutan pembahasan

- 01 Single-row dan kontrak jumlah baris
- 02 Multi-row dengan IN, ANY dan ALL
- 03 Nested dan correlated subquery
- 04 NULL dan jebakan NOT IN
- 05 UNION, UNION ALL, INTERSECT dan MINUS
- 06 CTE WITH dan pemeriksaan bertahap

Gambaran konsep dalam satu tabel

Pola	Pertanyaan yang dijawab	Perhatian
Single-row	Bandingkan dengan satu nilai	Kardinalitas dan kemungkinan NULL
IN / ANY / ALL	Bandingkan dengan kumpulan nilai	Makna operator dan nullability
EXISTS / NOT EXISTS	Ada atau tidak ada pasangan	Korelasi ke query luar
Set operators	Gabungkan / bandingkan hasil SELECT	Jumlah kolom dan tipe kompatibel
WITH / CTE	Beri nama tahap query	Baca dan uji tiap tahap

01 Single-row dan kontrak jumlah baris

Subquery adalah **SELECT** yang berada di dalam pernyataan lain.

Single-row subquery digunakan ketika ekspresi membutuhkan satu nilai, misalnya harga produk dibandingkan dengan **AVG** (harga) seluruh katalog. Aggregate tanpa **GROUP BY** menghasilkan satu baris, tetapi nilainya dapat **NULL** ketika data kosong.

POIN PEMBAHASAN

- Scalar subquery biasa yang tidak menemukan baris bernilai `NULL` ; yang menghasilkan beberapa baris menyebabkan kesalahan, bukan memilih baris pertama.
- Bedakan perilaku ini dari `SELECT INTO` pada PL/SQL yang dibahas kemudian.
- Gunakan kondisi primary key ketika membutuhkan satu produk.
- Jangan menambahkan batas baris semata untuk menyembunyikan pertanyaan yang sebenarnya mengharuskan banyak hasil atau menutupi masalah kualitas data.

02 Multi-row dengan IN, ANY dan ALL

Multi-row subquery mengembalikan kumpulan nilai.

IN menguji apakah ID termasuk dalam kumpulan, cocok untuk pelanggan yang pernah bertransaksi. `ANY` memerlukan setidaknya satu perbandingan `TRUE` , sedangkan `ALL` membutuhkan semua perbandingan `TRUE` .

POIN PEMBAHASAN

- Untuk kumpulan harga positif tanpa `NULL` , harga > `ANY` berarti lebih tinggi dari setidaknya satu harga, bukan lebih tinggi dari rata-rata.
- Harga > `ALL` berarti melebihi setiap harga di kumpulan itu.
- `ANY` terhadap kumpulan kosong bernilai `FALSE` ; `ALL` terhadap kumpulan kosong bernilai `TRUE` .
- Jika kumpulan mengandung `NULL` , logika `UNKNOWN` juga berperan.
- Contoh memakai harga produk yang `NOT NULL` sehingga pembaca dapat memusatkan perhatian pada arti operator.

03 Nested dan correlated subquery

Nested subquery menempatkan satu subquery di dalam subquery lainnya.

Pertanyaan produk yang dibeli pelanggan dari Subang dapat dibaca dari dalam: cari pelanggan kota tersebut, cari kepala penjualannya, lalu cari ID produk pada detail. Correlated subquery merujuk alias query luar.

POIN PEMBAHASAN

- `EXISTS` memeriksa keberadaan setidaknya satu baris yang sesuai untuk pelanggan saat ini; isi `SELECT 1` bukan nilai yang ditampilkan pada laporan.
- Pola ini menjelaskan kebutuhan tanpa menghasilkan satu baris pelanggan per detail pembelian.
- Secara konseptual hubungan diuji per baris luar, tetapi jangan menyatakan Oracle pasti mengeksekusi query secara berulang dengan cara tertentu: optimizer dapat mentransformasi rencana sesuai keadaan.

04 NULL dan jebakan NOT IN

Seed mempunyai penjualan anonim dengan `id_pelanggan NULL`.

`NOT IN` atas seluruh kolom tersebut tidak mengembalikan pelanggan tanpa penjualan karena perbandingan terhadap `NULL` dapat menjadi `UNKNOWN`. `WHERE` hanya menerima `TRUE`.

POIN PEMBAHASAN

- Ini bukan berarti semua pelanggan telah membeli.
- Dua perbaikan yang dapat diuji adalah menyaring `NULL` dalam subquery atau memakai `NOT EXISTS` yang dikorelasikan ke primary key pelanggan.
- `NOT EXISTS` menanyakan apakah tidak ada baris dengan ID sama; baris anonim tidak cocok dengan ID pelanggan yang non-`NULL`.
- Pada Oracle, string karakter kosong juga diperlakukan sebagai `NULL`.
- Gunakan `IS NULL` dan `IS NOT NULL`, bukan `= NULL`, lalu tulis asumsi nullability sebelum memakai operator himpunan atau perbandingan.

05 UNION, UNION ALL, INTERSECT dan MINUS

Set operator menumpuk hasil **SELECT** yang mempunyai jumlah kolom sama dan kelompok tipe data kompatibel pada posisi yang bersesuaian.

UNION menggabungkan dan menghapus duplikasi; **UNION ALL** mempertahankannya. INTERSECT menghasilkan nilai unik yang terdapat pada kedua hasil.

POIN PEMBAHASAN

- MINUS menghasilkan nilai unik dari hasil pertama yang tidak terdapat pada hasil kedua, sehingga urutan operand berpengaruh.
- Operasi ini bukan pengganti JOIN yang menambahkan kolom dari pasangan baris.
- Beri alias pada hasil pertama dan tempatkan **ORDER BY** pada akhir compound query.
- Contoh memakai ID produk dari katalog dan detail, sehingga Sabun yang belum mempunyai detail mudah diidentifikasi.
- Untuk operasi campuran, gunakan pengelompokan eksplisit agar maksud tidak bergantung pada asumsi prioritas.

06 CTE WITH dan pemeriksaan bertahap

CTE **WITH** memberi nama pada query sementara di dalam satu pernyataan.

CTE total_detail merangkum nilai item per transaksi, lalu query utama menggabungkan ringkasan itu dengan kepala penjualan dan pelanggan. Ini meningkatkan keterbacaan, bukan otomatis menciptakan tabel atau menjamin hasil disimpan secara fisik.

POIN PEMBAHASAN

- Uji dahulu isi CTE sebagai **SELECT** mandiri; kemudian periksa baris anonim dan transaksi tanpa detail dengan LEFT JOIN.
- **NVL** (total, 0) tepat ketika tidak adanya detail didefinisikan sebagai total nol pada laporan.
- Semua contoh hanya membaca data dan dapat dijalankan pada XE lokal atau FreeSQL schema pribadi setelah setup.

- Catat hasil aktual dan perubahan seed; jangan menganggap prediksi contoh sebagai bukti telah mengeksekusi Oracle.

Alur membangun query lanjutan

- 1 Nyatakan pertanyaan bisnis
- 2 Uji subquery sendiri
- 3 Periksa jumlah baris dan `NULL`
- 4 Gabungkan dengan query utama
- 5 Bandingkan hasil dengan seed

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.

Pelanggan yang belum membeli dan jebakan NULL

Hasil subquery ID pelanggan pada penjualan adalah 1, 2 dan NULL.

Contoh SQL / PL/SQL

```
SELECT p.id_pelanggan, p.nama
FROM pelanggan_demo p
WHERE NOT EXISTS (SELECT 1
FROM penjualan_demo j
```

```
WHERE j.id_pelanggan=p.id_pelanggan)
ORDER BY p.id_pelanggan;
```

Predikat	Jumlah hasil	Pelanggan
NOT IN dengan hasil berisi NULL	0	Tidak ada
NOT EXISTS yang berkorelasi	1	Citra

Cara membaca contoh

1. NOT IN membandingkan kandidat dengan seluruh daftar, termasuk NULL ; kondisi nonmatch menjadi UNKNOWN .
2. WHERE tidak mempertahankan UNKNOWN sehingga Citra juga tidak lolos NOT IN tersebut.
3. NOT EXISTS hanya memeriksa ada/tidaknya pasangan yang cocok, sehingga Citra ditemukan.

Yang perlu dipahami: Bukan datanya hilang: pilihan predikat dan penanganan NULL menentukan hasil.

Video pembelajaran

Video utama — fokus bab ini

Subquery Oracle dan single-row subquery. Referensi tambahan membahas correlated subquery/set operators dan WITH/CTE.

Mastering Oracle Subqueries (Part 1) - A detailed practical session for beginners.

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

Video tambahan sesuai subtopik

1. [Day - 06 - Oracle SQL SET Operations, Union, Union All, Intersect, Minus, Analytical functions](#)

NIC IT ACADEMY

Fokus: Correlated subquery dan Oracle UNION, UNION ALL, INTERSECT, MINUS.

Fokus bagian 00:16–33:00; pembahasan analytical functions sesudahnya bersifat opsional.

2. [How to Use SQL Common Table Expressions \(CTE\) in Oracle and SQL Server](#)

Sean MacKenzie Data Engineering

Fokus: WITH/Common Table Expressions, termasuk penerapan pada Oracle.

Video membandingkan Oracle dan SQL Server; gunakan syntax WITH Oracle pada halaman.

Batas cakupan dan panduan menonton: Video utama sebelumnya memakai SQL Server. ANY/ALL, nested/multiple-row subquery dan jebakan NULL pada NOT IN mengikuti kode Oracle dalam praktikum; tidak semua subtopik tercakup di satu video.

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 — Single-row dan multi-row IN

```
SELECT id_produk, nama_produk, harga
FROM produk_demo
WHERE harga > (SELECT AVG(harga) FROM produk_demo)
ORDER BY id_produk;

SELECT id_pelanggan, nama
FROM pelanggan_demo
WHERE id_pelanggan IN (SELECT id_pelanggan FROM penjualan_demo)
ORDER BY id_pelanggan;
```

AVG memberikan satu nilai; IN menerima banyak ID. NULL pada IN tidak menjadi kecocokan bagi pelanggan.

Langkah 2 — ANY dan ALL

```
SELECT id_produk, nama_produk, harga
FROM produk_demo
WHERE harga > ANY (SELECT harga FROM produk_demo WHERE stok > 0)
ORDER BY id_produk;

SELECT id_produk, nama_produk, harga
FROM produk_demo
WHERE harga > ALL (SELECT harga FROM produk_demo WHERE id_produk
IN (12, 13))
ORDER BY id_produk;
```

Bandingkan setiap harga dengan kumpulan yang dipilih. Ubah filter menjadi WHERE 1 = 0 untuk mengamati perbedaan himpunan kosong.

Langkah 3 — Nested subquery dan correlated EXISTS

```
SELECT id_produk, nama_produk
FROM produk_demo
WHERE id_produk IN (
  SELECT id_produk FROM detail_jual_demo
  WHERE id_jual IN (
    SELECT id_jual FROM penjualan_demo
    WHERE id_pelanggan IN (
      SELECT id_pelanggan FROM pelanggan_demo WHERE kota = 'Subang'
    )
  )
)
ORDER BY id_produk;

SELECT p.id_pelanggan, p.nama
FROM pelanggan_demo p
WHERE EXISTS (
  SELECT 1 FROM penjualan_demo j
  WHERE j.id_pelanggan = p.id_pelanggan AND j.status = 'LUNAS'
)
ORDER BY p.id_pelanggan;
```

Seed menghubungkan Subang dengan Ayu dan Kopi. EXISTS kedua memeriksa penjualan lunas setiap pelanggan tanpa duplikasi pelanggan.

Langkah 4 — NOT IN yang bermasalah dan dua perbaikannya

```
-- Sengaja untuk menunjukkan jebakan NULL; query ini tetap valid.
SELECT id_pelanggan, nama FROM pelanggan_demo
WHERE id_pelanggan NOT IN (SELECT id_pelanggan FROM
  penjualan_demo)
ORDER BY id_pelanggan;

SELECT id_pelanggan, nama FROM pelanggan_demo
WHERE id_pelanggan NOT IN (
  SELECT id_pelanggan FROM penjualan_demo WHERE id_pelanggan IS NOT
```

```

NULL
)
ORDER BY id_pelanggan;

SELECT p.id_pelanggan, p.nama FROM pelanggan_demo p
WHERE NOT EXISTS (
  SELECT 1 FROM penjualan_demo j WHERE j.id_pelanggan =
p.id_pelanggan
)
ORDER BY p.id_pelanggan;

```

Prediksi seed: query pertama kosong, dua perbaikan menemukan Citra. Primary key pelanggan non-NULL membuat perbandingan kedua pendekatan sesuai.

Langkah 5 — Empat set operator pada produk

```

SELECT id_produk FROM produk_demo
UNION
SELECT id_produk FROM detail_jual_demo
ORDER BY id_produk;

SELECT id_produk FROM produk_demo
UNION ALL
SELECT id_produk FROM detail_jual_demo
ORDER BY id_produk;

SELECT id_produk FROM produk_demo
INTERSECT
SELECT id_produk FROM detail_jual_demo
ORDER BY id_produk;

SELECT id_produk FROM produk_demo
MINUS
SELECT id_produk FROM detail_jual_demo
ORDER BY id_produk;

```

UNION ALL mempertahankan pengulangan ID; INTERSECT menemukan produk pada detail; MINUS menemukan produk katalog tanpa detail. Tidak memakai EXCEPT yang bukan target 19c.

Langkah 6 — CTE WITH untuk laporan total

```
WITH total_detail AS (
  SELECT id_jual, SUM(jumlah * harga) AS total
  FROM detail_jual_demo GROUP BY id_jual
)
SELECT j.id_jual, NVL(p.nama, 'Anonim') AS pelanggan,
       NVL(t.total, 0) AS total
FROM penjualan_demo j
LEFT JOIN pelanggan_demo p ON p.id_pelanggan = j.id_pelanggan
LEFT JOIN total_detail t ON t.id_jual = j.id_jual
ORDER BY j.id_jual;
```

CTE berlaku hanya pada pernyataan ini. LEFT JOIN mempertahankan kepala penjualan yang belum memiliki detail.

Tugas / latihan

1. Pada XE ATAU FreeSQL schema pribadi, tulis single-row, nested dan correlated subquery untuk pertanyaan penjualan berbeda; sertakan alasan pemilihan operator.
2. Jalankan tiga query NOT IN/NOT EXISTS dan jelaskan hasil berdasarkan NULL pada transaksi anonim. Jangan menghapus baris anonim untuk memperbaiki query.
3. Bandingkan empat set operator, lalu buat laporan CTE total per pelanggan termasuk yang belum membeli. Serahkan SQL dan hasil aktual.

Pertanyaan analisis

1. Apa perbedaan scalar subquery tanpa hasil dan SELECT INTO tanpa hasil pada PL/SQL? Nyatakan kontrak konteks masing-masing.
2. Apakah mengganti UNION ALL dengan UNION selalu meningkatkan kualitas data? Jelaskan kemungkinan hilangnya pengulangan yang sah.

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 — Using Subqueries](#)
- [Oracle 19c — IN Condition](#)
- [Oracle 19c — Comparison Conditions](#)
- [Oracle 19c — Set Operators](#)
- [Oracle 19c — SELECT dan WITH](#)

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