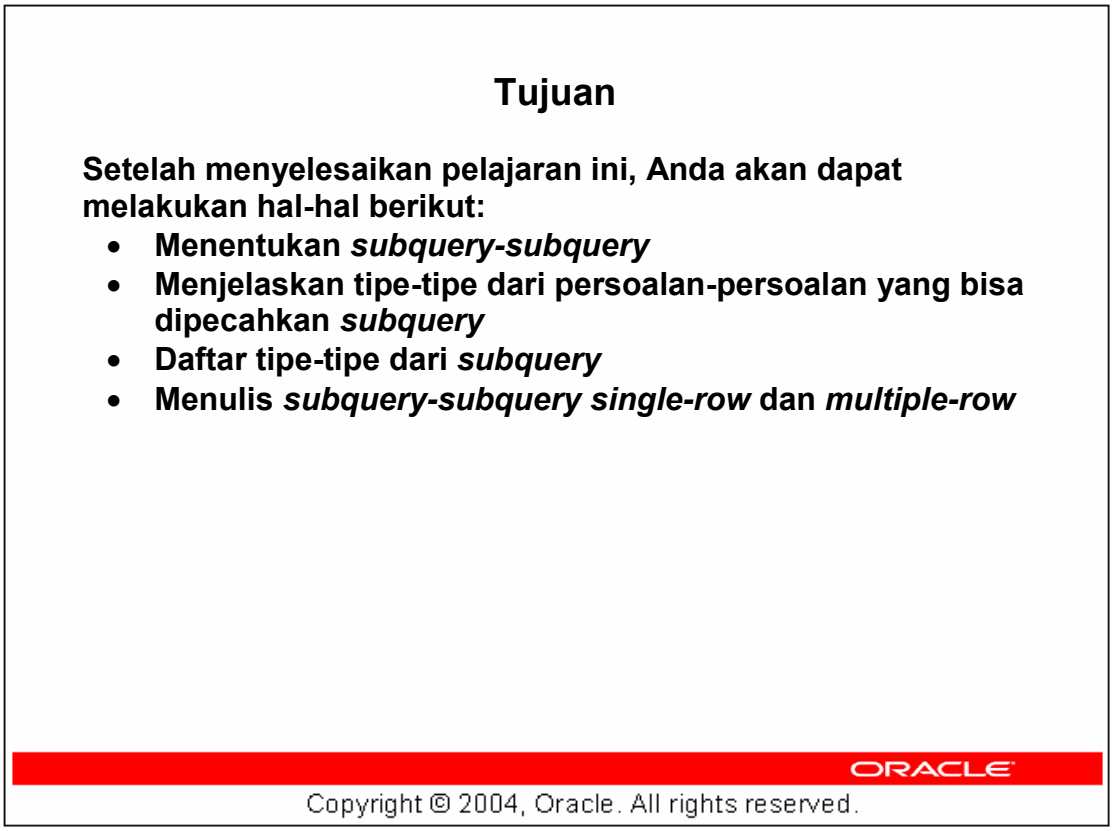
6

Menggunakan *Subquery* untuk Memecahkan Query-Query

ORACLE

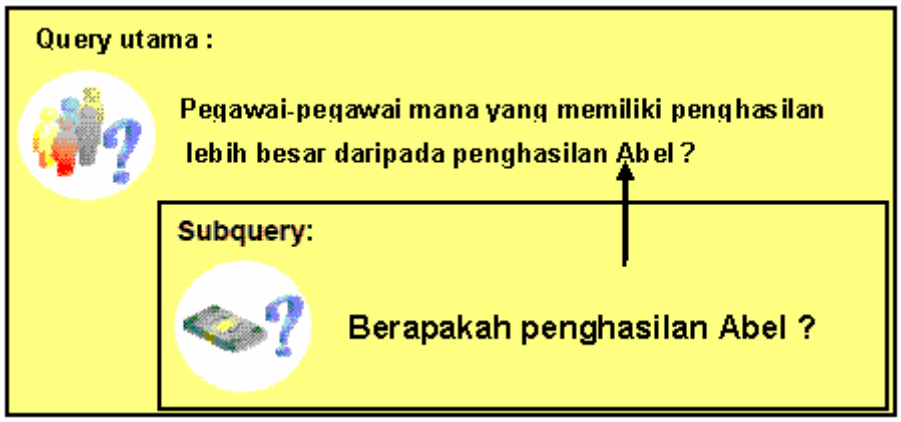


Tujuan

Dalam pelajaran ini, Anda belajar tentang fitur-fitur lebih lanjut dari pernyataan `SELECT`. Anda dapat menulis *subquery-subquery* pada klausa `WHERE` dari pernyataan SQL yang lain untuk memperoleh nilai-nilai berdasarkan pada suatu nilai kondisional yang tidak diketahui. Pelajaran ini meliputi *single-row subquery* dan *multiple-row subquery*.

Menggunakan Suatu *Subquery* untuk Memecahkan Suatu Persoalan

Siapakah yang memiliki penghasilan lebih besar daripada penghasilan Abel ?



ORACLE

Menggunakan suatu *Subquery* untuk Memecahkan suatu Persoalan

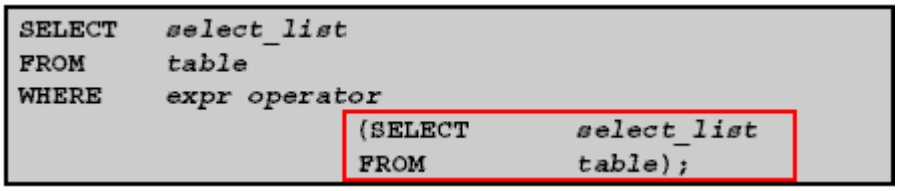
Misalkan Anda ingin menulis suatu query untuk mencari tahu penghasilan siapa yang lebih besar daripada penghasilan Abel.

Untuk memecahkan masalah ini, Anda memerlukan *dua* query: satu query untuk mencari berapa banyak penghasilan Abel, dan query kedua untuk mencari penghasilan siapa yang lebih besar dari jumlah itu.

Anda dapat memecahkan persoalan ini dengan menggabungkan dua query, menempatkan satu query *di dalam* query lain.

Inner query (atau *subquery*) mengembalikan suatu nilai yang digunakan *outer query* (atau query utama). Penggunaan suatu *subquery* sama dengan penggunaan dua query berturut-turut dan menggunakan hasil dari query pertama sebagai nilai pencari dalam query yang kedua.

Sintak *Subquery*



- ***Subquery* (*inner query*) dieksekusi sekali sebelum query utama (*outer query*).**
- **Hasil dari *subquery* digunakan oleh query utama.**

Sintak *Subquery*

Suatu *subquery* adalah suatu pernyataan `SELECT` yang dilekatkan didalam suatu klausa pada pernyataan `SELECT` lain. Anda dapat membangun pernyataan-pernyataan yang kuat selain yang sederhana dengan menggunakan *subquery-subquery*. *Subquery-subquery* bisa sangat bermanfaat ketika Anda memerlukan untuk memilih baris-baris dari suatu table dengan suatu kondisi yang tergantung pada data didalam tabel itu sendiri.

Anda dapat menempatkan *subquery* didalam sejumlah klausa-klausa SQL, termasuk berikut :

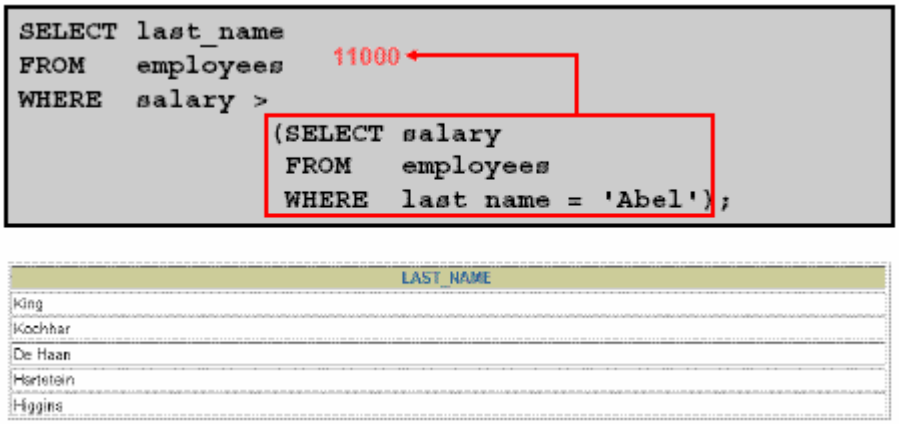
- Klausa `WHERE`
- Klausa `HAVING`
- Klausa `FROM`

Didalam Syntax :

`Operator` termasuk suatu kondisi pembanding seperti >, =, atau `IN`
Catatan : Kondisi-kondisi pembanding dibagi dalam dua kelas : *single-row operator* (>,=,>=,<,<>,<=) dan *multiple-row operator* (`IN`, `ANY`, `ALL`).

Subquery lebih dikenal sebagai suatu `SELECT` bersarang (*nested*), *sub*-`SELECT`, atau pernyataan *inner* `SELECT`. Secara umum s*ubquery* dieksekusi pertama kali, dan hasilnya digunakan untuk melengkapi kondisi query pada query utama (atau *outer*).

Menggunakan suatu *Subquery*



Menggunakan suatu *Subquery*

Dalam slide, *inner query* menentukan penghasilan dari pegawai bernama Abel. *Outer query* mengambil hasil dari *inner query* dan menggunakan hasil ini untuk menampilkan semua pegawai yang berpenghasilan lebih dari jumlah tersebut.

Pedoman-Pedoman untuk Menggunakan Subquery

- ***Subquery-subquery* diapit tanda kurung.**
- **Tempatkan *subquery-subquery* di sebelah kanan dari kondisi pembanding.**
- **Klausa `ORDER BY` dalam *subquery* tidak diperlukan kecuali Anda melakukan pemeringkatan (*Top-N analysis*).**
- **Gunakan *single-row operator* pada *single-row subquery*, dan gunakan *multiple-row operator* pada *multiple-row subquery*.**

ORACLE

Pedoman-Pedoman untuk Menggunakan *Subquery*

- Suatu *subquery* harus diapit tanda kurung.
- Tempatkan s*ubquery* disisi kanan dari kondisi pembanding agar mudah dibaca.
- Pada Oracle8*i* dan keluaran-keluaran berikutnya, suatu klausa `ORDER BY` bisa digunakan dan diperlukan dalam *subquery* untuk melakukan pemeringkatan (*Top-N analysis*).
  - Sebelumnya pada Oracle8*i*, bagaimanapun, *subquery-subquery* tidak bisa memuat klausa `ORDER BY`. Klausa `ORDER BY` hanya sekali digunakan untuk suatu pernyataan `SELECT;` jika ditentukan, klausa `ORDER BY` berada diakhir pada pernyataan `SELECT` utama.
- Dua kelas dari kondisi-kondisi pembanding digunakan dalam *subquery-subquery*: *single-row operator* dan *multiple-row operator*.

Tipe-Tipe dari Subquery

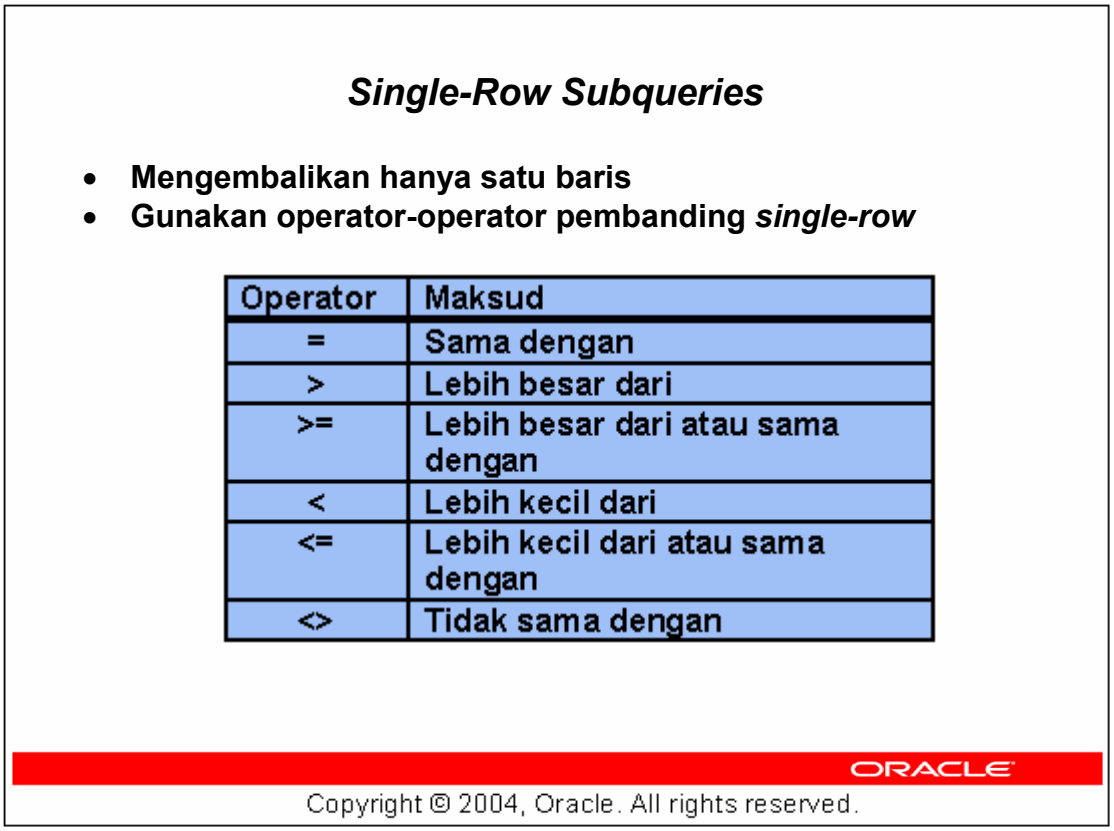
- ***Single-row subquery***

- ***Multiple-row subquery***

Tipe-Tipe dari *Subquery*

- *Single-row subquery*: Query yang mengembalikan hanya satu baris dari pernyataan *inner* `SELECT (SELECT` terdalam).
- *Multiple-row subquery*: Query yang mengembalikan lebih dari satu baris dari pernyataan *inner* `SELECT`.

Catatan : Ada juga *multiple-column subquery*, dimana query-query itu mengembalikan lebih dari satu kolom dari pernyataan *inner* `SELECT`. Hal tersebut dicakup dalam *Oracle Database 10g*: *SQL Fundamentals II Course*.



Single-Row Subqueries

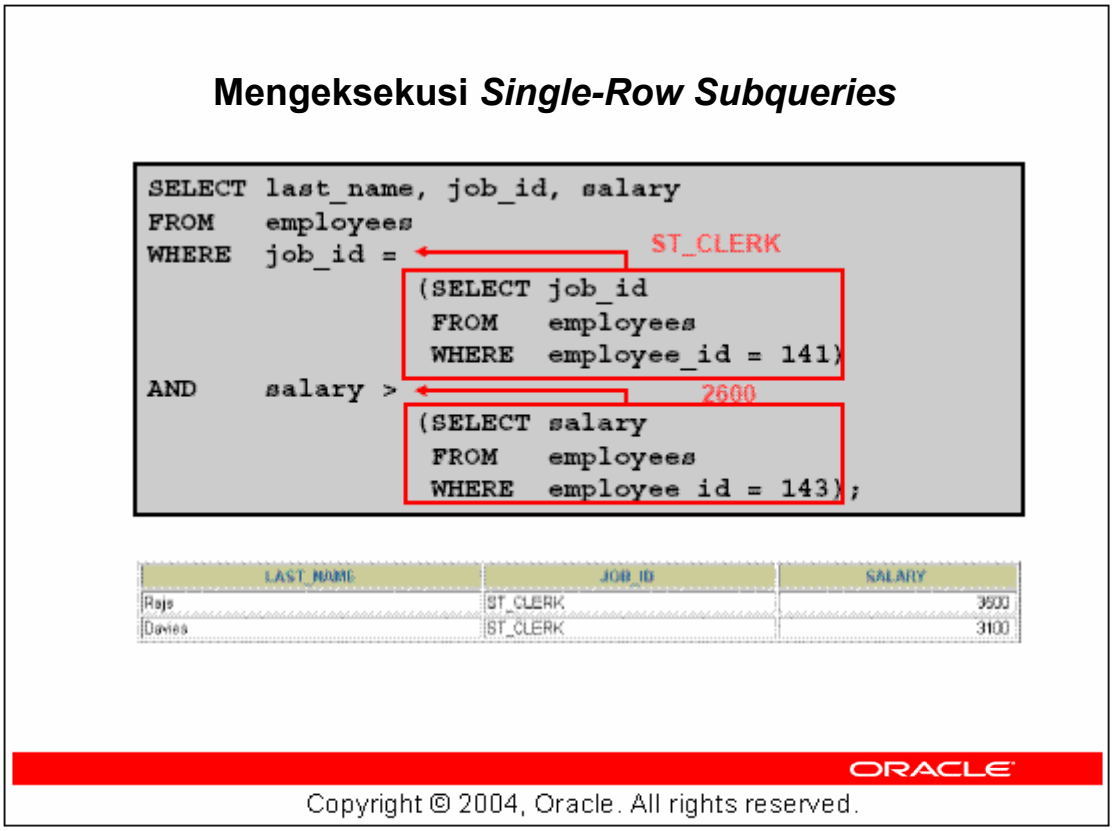
Suatu *single-row subquery* adalah mengembalikan satu baris dari pernyataan *inner* `SELECT`. Tipe dari *subquery* ini menggunakan suatu *single-row operator*. Pada slide diberikan suatu daftar dari operator-operator *single-row*.

Contoh

Tampilkan pegawai-pegawai yang job ID-nya sama dengan pegawai 141:

```
SELECT last_name, job_id
FROM   employees
WHERE  job_id  =
                  (SELECT job_id
                   FROM employees
                   WHERE employee_id  = 141 );
```

LAST_NAME	JOB_ID
Rajs	ST_CLERK
Davies	ST_CLERK
Matos	ST_CLERK
Vargas	ST_CLERK



Mengeksekusi *Single-Row Subqueries*

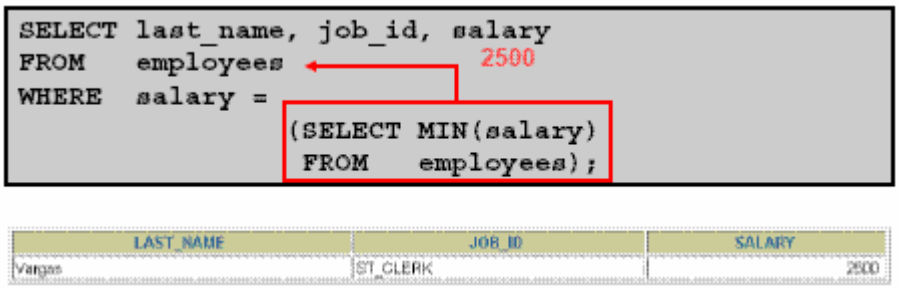
Pernyataan `SELECT` bisa dianggap sebagai suatu blok query. Contoh pada slide menampilkan pegawai-pegawai yang job ID-nya sama dengan pegawai 141 dan siapa saja penghasilannya lebih besar dari pegawai 143.

Pada contoh terdapat tiga blok query: *outer query* dan dua *inner query*. Blok *inner query* dieksekusi pertama kali, secara berturut-turut menghasilkan hasil query `ST_CLERK` dan `2600`. Blok *outer query* kemudian memproses dan menggunakan nilai-nilai yang dikembalikan oleh *inner queries* untuk melengkapi kondisi-kondisi pencariannya.

Dua *inner query* mengembalikan nilai-nilai tunggal (secara berturut-turut `ST_CLERK` dan `2600`), jadi pernyataan SQL ini disebut suatu *single-row subquery*.

Catatan : *Outer query* dan *inner query* bisa mengambil data dari tabel-tabel yang berbeda.

Menggunakan *Group Functions* dalam suatu *Subquery*



LAST_NAME	JOB_ID	SALARY
Vargas	ST_CLERK	2500

ORACLE

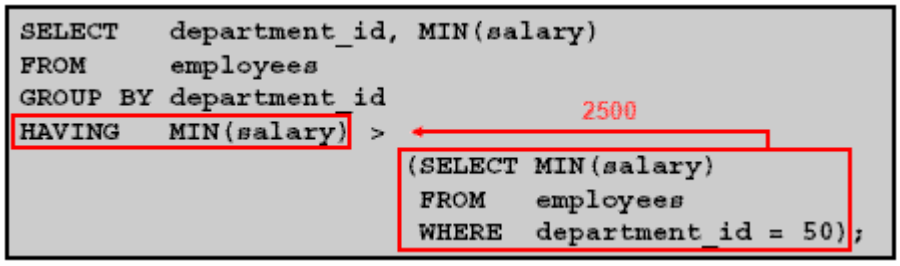
Menggunakan *Grup Function* dalam suatu *Subquery*

Anda bisa menampilkan data dari suatu query utama dengan menggunakan *grup function* didalam *subquery* untuk mengembalikan suatu baris tunggal. *Subquery* berada dalam tanda kurung dan diletakkan setelah kondisi pembanding.

Contoh pada slide menampilkan nama belakang pegawai, job ID, dan penghasilan seluruh pegawai yang penghasilannya sama dengan penghasilan minimum. *Group function* `MIN` mengembalikan suatu nilai tunggal (`2500`) untuk *outer query*.

Klausa HAVING pada Subquery-Subquery

- **Server Oracle pertama mengeksekusi *subquery-subquery*.**
- **Server Oracle mengembalikan hasil-hasil kedalam klausa HAVING dari query utama.**



ORACLE

Klausa HAVING dengan *Subquery*

Anda dapat menggunakan *subquery-subquery* tidak hanya pada klausa WHERE tapi bisa juga pada klausa HAVING. Server Oracle mengeksekusi *subquery*, dan hasilnya dikembalikan ke dalam klausa HAVING pada query utama.

Pernyataan SQL dalam slide menampilkan semua departemen yang mempunyai penghasilan minimum lebih besar dari departemen 50.

DEPARTMEN_ID	MIN(SALARY)
10	4400
20	6000
. . .	
	7000

7 ROWS SELECTED

Contoh :
Cari jabatan dengan penghasilan rata-rata minimum.

```
SELECT      job_id, AVG(salary)
FROM        employees
GROUP BY    job_id
HAVING      AVG(salary) = (SELECT   MIN(AVG(salary))
                           FROM     employees
                           GROUP BY job_id);
```

Apa yang Salah pada Pernyataan Ini ?

```
SELECT employee_id, last_name
FROM   employees
WHERE  salary =
                (SELECT   MIN(salary)
                 FROM     employees
                 GROUP BY department_id);
```

```
ERROR at line 4:
ORA-01427: single-row subquery returns more than
one row
```

Single-row operator* pada *multiple-row subquery

Kesalahan-Kesalahan pada Subquery-Subquery

Satu kesalahan umum pada subquery terjadi ketika lebih dari satu baris dikembalikan untuk suatu *single-row subquery*.

Pada pernyataan SQL dalam slide, *subquery* berisi suatu klausa `GROUP BY`, yang berakibat subquery itu akan mengembalikan banyak baris, satu dari setiap kelompok yang ditemukannya. Dalam kasus ini, hasil dari *subquery* adalah `4400`, `6000`, `2500`, `4200`, `7000`, `17000`, dan `8300`.

Outer query mengambil hasil-hasil itu dan menggunakannya pada klausa `WHERE`. Klausa `WHERE` berisi suatu operator samadengan (=), suatu operator pembanding *single-row* yang dikira hanya satu nilai. Operator = tidak menerima lebih dari satu nilai dari *subquery* dan oleh karena itu menghasilkan kesalahan.

Untuk memperbaiki kesalahan ini, rubah operator = menjadi `IN`.

Akankah Pernyataan ini Mengembalikan Baris-Baris ?

```
SELECT last_name, job_id
FROM   employees
WHERE  job_id =
                (SELECT job_id
                 FROM   employees
                 WHERE  last_name = 'Haas');
```

```
no rows selected
```

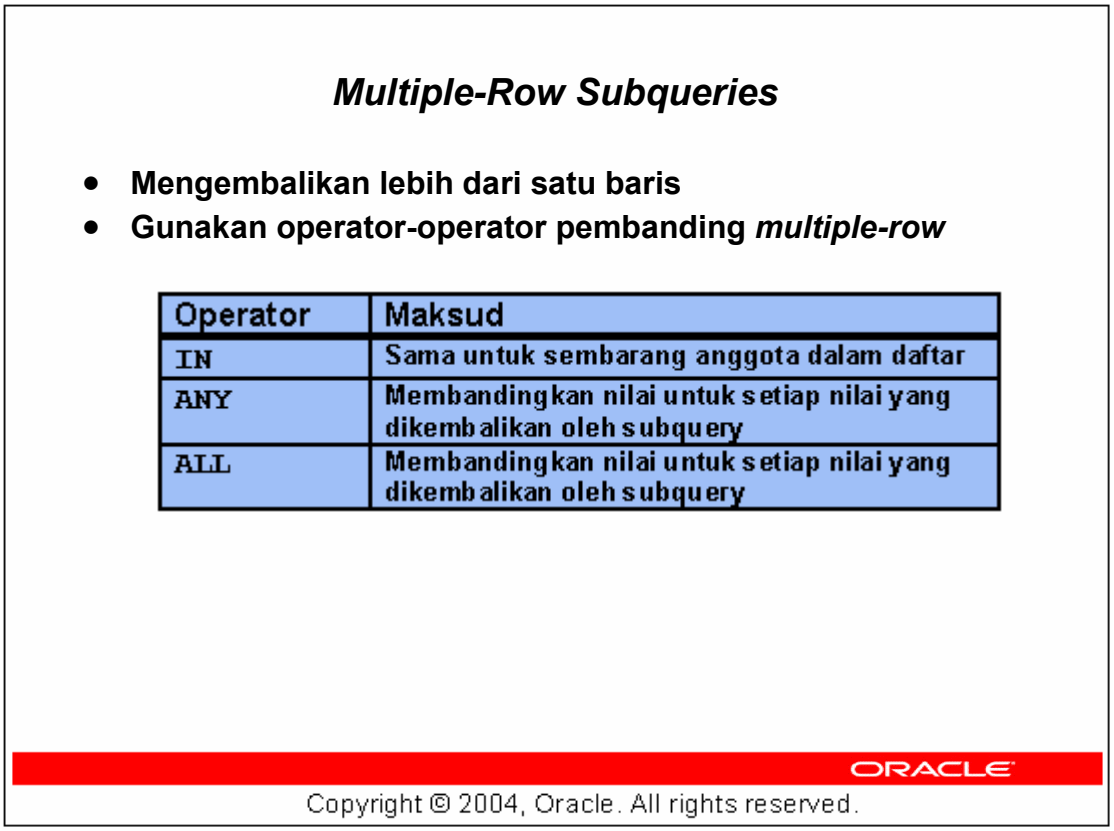
Subquery tidak mengembalikan nilai-nilai.

ORACLE

Persoalan-Persoalan pada Subquery-Subquery

Suatu persoalan umum pada subquery-subquery muncul saat tidak ada baris-baris yang dikembalikan oleh *inner query*. Pada pernyataan SQL dalam slide, *subquery* berisi suatu klausa `WHERE`. Rupanya, tujuannya adalah untuk mencari pegawai yang bernama Haas. Pernyataannya benar tetapi tidak ada baris yang terpilih ketika dieksekusi.

Tidak ada pegawai bernama Haas. Sehingga *subquery* tidak mengembalikan baris-baris. *Outer query* mengambil hasil-hasil dari *subquery* (*null*) dan menggunakan hasil-hasil tersebut pada klausa `WHERE` *outer query*. *Outer query* tidak mencari pegawai dengan suatu job ID sama dengan *null*, jadi tidak mengembalikan baris-baris. Jika suatu job berada pada suatu nilai *null*, baris tidak dikembalikan karena perbandingan dua nilai *null* menghasilkan suatu *null*. Oleh karena, kondisi `WHERE` tidak benar.



Multiple-Row Subqueries

Subquery-subquery yang mengembalikan lebih dari satu baris disebut *multiple-row subqueries*. Anda menggunakan suatu *multiple-row operator*, disamping suatu *single-row operator*, pada suatu *miltiple-row subquery*. *Multiple-row operator* memperkirakan satu atau lebih nilai-nilai :

Contoh
Cari pegawai-pegawai yang mendapat penghasilan yang sama dengan penghasilan minimum untuk setiap departemen.

Inner query dieksekusi pertama kali, menghasilkan suatu hasil query. Blok query utama kemudian memproses dan menggunakan nilai-nilai yang dikembalikan oleh *inner query* untuk melengkapi kondisi pencariannya. Sesungguhnya, query utama tampak pada server Oracle sebagai berikut:

```
SELECT last_name, salary, department_id
FROM   employess
WHERE  salary IN (2500, 4200, 6000, 7000, 800, 8600, 17000);
```

Menggunakan Operator ANY dalam *Multiple-Row Subqueries*

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary < ANY
                (SELECT salary
                 FROM   employees
                 WHERE  job_id = 'IT_PROG')
AND    job_id <> 'IT_PROG';
```

9000, 6000, 4200

EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
124	Mourgos	ST_MAN	5800
141	Rajs	ST_CLERK	3500
142	Davies	ST_CLERK	3100
143	Matos	ST_CLERK	2600
144	Vargas	ST_CLERK	2500

...

10 rows selected.

ORACLE

Multiple-Row Subqueries (lanjutan)

Operator ANY (dan sinonimnya, operator SOME) membandingkan suatu nilai pada *setiap* nilai yang dikembalikan oleh suatu *subquery*. Contoh pada slide menampilkan para pegawai yang bukan IT programmers dan penghasilan siapa yang kurang dari beberapa IT programmer. Penhasilan maksimum yang didapat seorang programmer adalah $ 9,000.

<ANY maksudnya kurang dari maksimum. >ANY maksudnya lebih dari minimum. =ANY adalah sama dengan IN.

Menggunakan Operator ALL dalam *Multiple-Row Subqueries*

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary < ALL
                (SELECT salary
                 FROM   employees
                 WHERE  job_id = 'IT_PROG')
AND    job_id <> 'IT_PROG';
```

9000, 6000, 4200

EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
141	Rajs	ST_CLERK	3500
142	Davies	ST_CLERK	3100
143	Matos	ST_CLERK	2900
144	Vargas	ST_CLERK	2500

ORACLE

Multiple-Row Subqueries (lanjutan)

Operator `ALL` membandingkan suatu nilai *untuk setiap* nilai yang dikembalikan oleh suatu *subquery*. Contoh dalam slide menampilkan para pegawai yang penghasilannya kurang dari penghasilan dari semua pegawai dengan suatu job ID `IT_PROG` dan siapa yang bukan `IT_PROG`.

`>ALL` maskudnya lebih dari maksimum, dan `<ALL` maksudnya kurang dari minimum.

Operator `NOT` dapat digunakan dengan operator-operator `IN`, `ANY`, dan `ALL`.

Nilai-Nilai *NULL* dalam suatu *Subquery*

```
SELECT emp.last_name
FROM   employees emp
WHERE  emp.employee_id NOT IN
                           (SELECT mgr.manager_id
                            FROM   employees mgr);

no rows selected
```

ORACLE

Mengembalikan *Null-Null* dalam Menghasilkan Kumpulan dari suatu *Subquery*

Pernyataan SQL dalam slide mencoba untuk menampilkan semua pegawai yang tidak memiliki beberapa subordinate. Secara logika, pernyataan SQL ini akan dikembalikan 12 baris. Bagaimanapun, pernyataan SQL tidak mengembalikan beberapa baris pun. Satu dari nilai-nilai dikembalikan oleh *inner query* adalah nilai *null*, dan dan karenanya keseluruhan query tidak mengembalikan baris-baris.

Alasan adalah bahwa semua kondisi-kondisi yang membandingkan suatu nilai *null* menghasilkan suatu *null*. Jadi kapanpun nilai-nilai *null* secara kemungkinan menjadi bagian dari sekelompok hasil-hasil dari suatu *subquery*, jangan gunakan operator `NOT IN`. Operator `NOT IN` sama dengan operator `<>ALL`.

Sebagai catatan bahwa nilai *null* sebagai bagian dari sekelompok hasil-hasil dari suatu subquery adalah tidak menjadi masalah jika Anda menggunakan operator `IN`. Operator `IN` adalah sama dengan operator `=ANY`. Sebagai contoh, untuk menampilkan para pegawai yang memiliki subordinate, gunakan pernyataan SQL berikut :

```
SELECT emp.last_name
FROM   employees  emp
WHERE  emp.employees_id IN
                          (SELECT mgr.manager_id
                           FROM   employees  mgr);
```

Sebagai alternatif, suatu klausa `WHERE` dapat disertakan pada *subquery* untuk menampilkan seluruh pegawai yang tidak memiliki beberapa subordinate :

```
SELECT last_name FROM employees
WHERE  employee_id NOT IN
                         (SELECT manager_id
                          FROM   employees
                          WHERE  manager_id IS NOT NULL);
```

Ringkasan

Dalam pelajaran ini, Anda sudah mempelajari bagaimana untuk :

- **Mengenali kapan suatu *subquery* dapat membantu memecahkan suatu pertanyaan**
- **Menulis *subquery-subquery* ketika suatu query didasarkan pada nilai-nilai yang tidak diketahui**

```
SELECT   select_list
FROM     table
WHERE    expr operator
                    (SELECT select_list
                    FROM    table);
```

ORACLE

Ringkasan

Dalam pelajaran ini, Anda sudah mempelajari bagaimana untuk menggunakan *subquery-subquery*. Suatu *subquery* adalah suatu pernyataan `SELECT` yang dilekatkan pada suatu klausa dari pernyataan SQL yang lain . Subquery-subquery berguna ketika suatu query didasarkan pada suatu criteria pencarian dengan nilai-nilai lebih jauh tidak dikenal.

Subquery mempunyai karakteristik sebagai berikut :

- Dapat melawatkan satu baris data ke suatu pernyataan utama yang berisi suatu *single-row operator,* seperti =, <>, >, >=, <, atau <=
- Dapat melewatkan beberapa baris data ke suatu pernyataan utama yang berisi suatu *multiple row operator*, seperti `IN`
- Adalah yang pertama kali diproses oleh server Oracle, setelah klausa `WHERE` atau `HAVING` menggunakan hasil-hasil.
- Dapat berisi *group functions.*