Sub Query – Set Operator

6

Menggunakan *Subquery* untuk Memecahkan Query-Query

What is a subquery ?

- In Oracle, a subquery is **a query within a query**. You can create subqueries within your SQL statements.
- These subqueries can reside in the **WHERE** clause, the **FROM** clause, or the **SELECT** clause.
- These subqueries can used in the **INSERT**, **UPDATE**, and **DELETE** clause.

Sintak *Subquery*

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

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

Tipe-Tipe dari Subquery

- *Single-row subquery*

- *Multiple-row subquery*

ORACLE

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

ORACLE

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

Implementasi Subquery

1. Clausa **WHERE** atau **HAVING**
2. Operator **IN** dan **NOT IN**
3. Clausa **FROM**
4. Operator **EXIST** dan NOT **EXIST**

Implementasi - Clausa **WHERE** atau **HAVING**

Menampilkan data employees yang memiliki gaji tertinggi

```
SELECT *
FROM employees
WHERE salary =
   (
       SELECT max(salary)
       FROM employees
   );
```

Menampilkan nama departemen dan rata-rata gajinya, untuk yang rata-ratanya diatas rata-rata gaji seluruh employees

```
SELECT department_name, Avg(salary)
FROM Employees e natural join Departments
GROUP BY department_name
HAVING avg(salary) >
    (
       SELECT avg(salary)
       FROM employees
    );
```

Implementasi - Operator **IN** dan **NOT IN**

Menampilkan nama employees dan gajinya, untuk employees yang PERNAH mutasi bagian

```
SELECT first_name||' '||last_name
NAME,salary
FROM Employees
WHERE employee_id IN
   (
       SELECT employee_id
       FROM job_history
   );
```

Menampilkan nama employees dan gajinya, untuk employees yang TIDAK pernah mutasi bagian

```
SELECT first_name||' '||last_name
NAME,salary
FROM Employees
WHERE employee_id NOT IN
   (
       SELECT employee_id
       FROM job_history
   );
```

Implementasi - Operator **EXIST** dan NOT **EXIST**

Menampilkan nama employees dan gajinya, untuk employees yang PERNAH mutasi bagian

```
SELECT first_name||' '||last_name
NAME,salary
FROM Employees e
WHERE EXISTS
(
    SELECT employee_id
    FROM job_history
    WHERE employee_id=e.employee_id
);
```

Menampilkan nama employees dan gajinya, untuk employees yang TIDAK pernah mutasi bagian

```
SELECT first_name||' '||last_name
NAME,salary
FROM Employees e
WHERE NOT EXISTS
(
    SELECT employee_id
    FROM job_history
    WHERE employee_id=e.employee_id
);
```

Implementasi - Clausa **FROM**

- Disebut juga Inline View

Menampilkan jumlah gaji dari masing-masing job_id

```
SELECT job_id, jumlah_gaji
FROM
    (
        SELECT department_id, job_id,Sum(salary) JUMLAH_GAJI
        FROM Employees
        GROUP BY department_id,job_id
        ORDER BY 1,2
    )
ORDER BY 1 desc;
```

Scalar Subquery ➔ Subquery dalam SELECT

Display the average salary of all employees alongside the salary for every employee.

```
SELECT ( SELECT avg(salary) FROM employees) AS avg_sal, salary
FROM employees;
```

Correlated Subquery

The highest paid employee in each department

```
SELECT *
FROM employees e1
WHERE e1.salary =
(
SELECT max(salary)
FROM employees e2
WHERE e1.department_id = e2.department_id
);
```

Multiple-Row Subqueries

- Mengembalikan lebih dari satu baris
- Gunakan operator-operator pembanding *multiple-row*

Operator	Maksud
IN	Sama untuk sembarang anggota dalam daftar
ANY	Membandingkan nilai untuk setiap nilai yang dikembalikan oleh subquery
ALL	Membandingkan nilai untuk setiap nilai yang dikembalikan oleh subquery

ORACLE

Menggunakan Operator ANY dalam *Multiple-Row Subqueries*

```
SELECT employee_id, last_name, job_id, salary
FROM   employees          9000, 6000, 4200
WHERE  salary < ANY
                (SELECT salary
                 FROM   employees
                 WHERE  job_id = 'IT_PROG')
AND    job_id <> 'IT_PROG';
```

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

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	2600
144	Vargas	ST_CLERK	2500

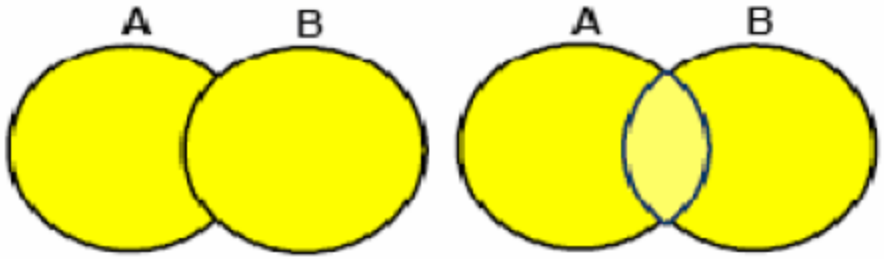
ORACLE

7

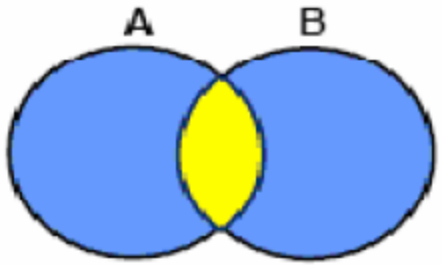
Menggunakan *Set Operators*

ORACLE

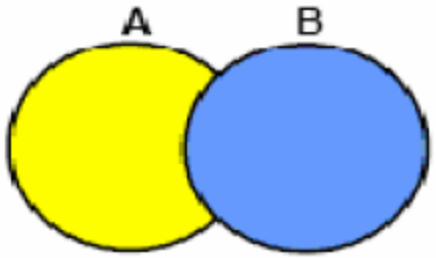
Set Operators



UNION/UNION ALL



INTERSECT



MINUS

ORACLE

Tabel-Tabel yang Digunakan dalam Pelajaran Ini

Tabel-tabel yang digunakan dalam pelajaran ini adalah :

- `EMPLOYEES` : Menyediakan rincian berdasarkan pegawai-pegawai saat ini
- `JOB_HISTORY` : Rincian catatan dari tanggal mulai dan tanggal berakhir dari suatu job yang terdahulu, dan nomor identifikasi job serta departemen saat seorang pegawai berpindah job.

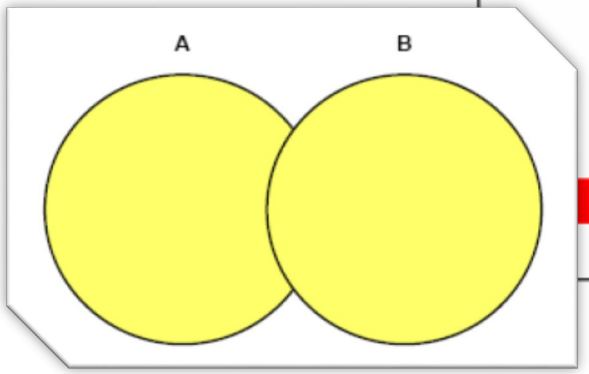
ORACLE

Menggunakan Operator UNION

Menampilkan detail pekerjaan dari semua karyawan baik yang sekarang atau sebelumnya. Menampilkan karyawan hanya sekali.

```
SELECT employee_id, job_id
FROM   employees
UNION
SELECT employee_id, job_id
FROM   job_history;
```

EMPLOYEE_ID	JOB_ID
100	AD_PRES
101	AC_ACCOUNT
...	
200	AC_ACCOUNT
200	AD_ASST
...	
205	AC_MGR
206	AC_ACCOUNT



ORACLE

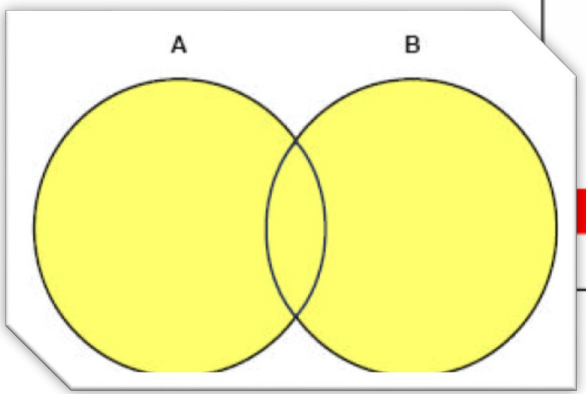
Menggunakan operator UNION ALL

Menampilkan semua karyawan yang bekerja di departemen baik yang sekarang maupun sebelumnya.

```
SELECT employee_id, job_id, department_id
FROM   employees
UNION ALL
SELECT employee_id, job_id, department_id
FROM   job_history
ORDER BY  employee_id;
```

EMPLOYEE_ID	JOB_ID	DEPARTMENT_ID
100	AD_PRES	90
101	AD_VP	90
...		
200	AD_ASST	10
200	AD_ASST	90
200	AC_ACCOUNT	90
...		
205	AC_MGR	110
206	AC_ACCOUNT	110

30 rows selected.



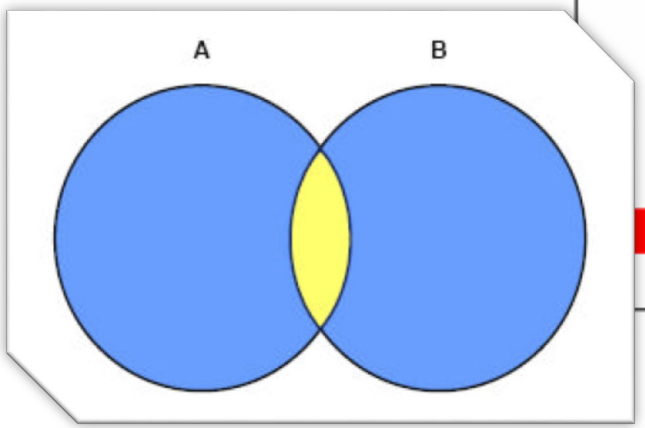
ORACLE

Menggunakan Operator INTERSECT

Menampilkan employee ID dan job ID dari tabel employee dimana karyawan tersebut mempunyai pekerjaan yang sama dengan pekerjaan mereka terdahulu (karyawan tersebut telah berganti pekerjaan tapi sekarang kembali lagi bekerja pada pekerjaan asal)

```
SELECT employee_id, job_id
FROM   employees
INTERSECT
SELECT employee_id, job_id
FROM   job_history;
```

EMPLOYEE_ID	JOB_ID
176	SA_REP
200	AD_ASST



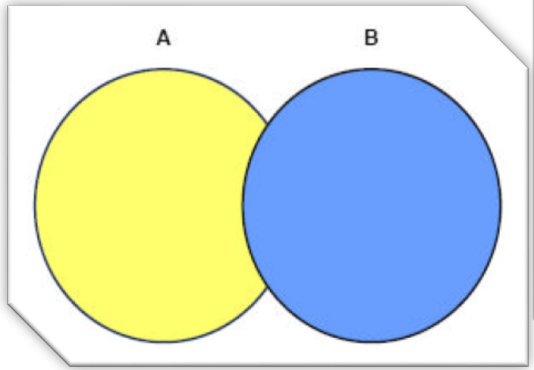
ORACLE

Operator MINUS

Menampilkan employee id dari karyawan-karyawan yang tidak pernah ganti pekerjaan sekalipun.

```
SELECT employee_id,job_id
FROM   employees
MINUS
SELECT employee_id,job_id
FROM   job_history;
```

EMPLOYEE_ID	JOB_ID
100	AD_PRES
101	AD_VP
102	AD_VP
103	IT_PROG
...	
201	MK_MAN
202	MK_REP
205	AC_MGR
206	AC_ACCOUNT



ORACLE

Petunjuk Set Operator

- Ekspresi yang terdapat pada `SELECT`list harus sama dengan nomor dan tipe datanya.
- Parentheses dapat digunakan untuk merubah sequence.
- `ORDER BY` clause :
  - Dapat digunakan hanya diakhir dari suatu statement.
  - Dapat menggunakan nama kolom, alias dari `SELECT` statement yang pertama, atau sesuai dengan letak penulisan.

ORACLE

Membandingkan Dengan Pernyataan SELECT

Menggunakan operator UNION, untuk menampilkan Department_ID, Location, dan Hire Date untuk semua karyawan.

```
SELECT department_id, TO_NUMBER(null)
       location, hire_date
FROM   employees
UNION
SELECT department_id, location_id,  TO_DATE(null)
FROM   departments;
```

DEPARTMENT_ID	LOCATION	HIRE_DATE
10	1700	
10		17-SEP-87
20	1800	
20		17-FEB-96
...		
110	1700	
110		07-JUN-94
190	1700	
		24-MAY-99

37 rows selected

Membandingkan Dengan Pernyataan SELECT : Contoh

Menggunakan operator UNION, untuk menampilkan employee_ID, job_id, dan salary untuk semua karyawan.

```
SELECT employee_id, job_id,salary
FROM   employees
UNION
SELECT employee_id, job_id,0
FROM   job_history;
```

EMPLOYEE_ID	JOB_ID	SALARY
100	AD_PRES	24000
101	AC_ACCOUNT	0
101	AC_MGR	0
...		
205	AC_MGR	12000
206	AC_ACCOUNT	8300

30 rows selected.

ORACLE


Semangat