

# **MI2154**

## **SQL LANJUT**

### **MODUL PRAKTIKUM**

Hanya dipergunakan di lingkungan Telkom Applied Science School



**Departemen Teknologi Informasi  
Telkom Applied Science School  
2014**

## **DAFTAR PENYUSUN**

1. Versi 1 : 2014 01 : RA. Paramita Mayadewi, S.Kom, M.T

## **Daftar Isi**

Daftar Penyusun.....	i
Daftar Isi.....	ii
1 Bab I Pengantar Controlling User Access.....	1
2 Bab II Schema Object.....	16
3 Bab III View .....	23
4 Bab IV Join Table .....	27
5 Bab V In Line View .....	36
6 Bab VI Merge.....	40
7 Bab VII Operator Set .....	43
8 Bab VIII Single Row SubQuery.....	47
9 Bab IX Multiple Row & Multiple Column Subquery .....	53
10 Bab X Scalar & Correlated Subquery.....	61
11 Daftar Pustaka .....	65

## 1 BAB I PENGANTAR CONTROLLING USER ACCESS

### 1.1 IDENTITAS

#### Kajian

Controlling User Access

#### Topik

1. Menciptakan User ,
2. Pengantar Hak Akses (*Privileges*)

#### Referensi

1. Gavin Powell, C.M.D.(2005). Oracle SQL Jumpstart With Examples. USA: Elsevier Inc.
2. Greenberg, N. (Edition 1.1 August 2004). Oracle Database 10g: SQL Fundamental II. Jobi Varghese.

#### Kompetensi Utama

1. Mampu menciptakan dan menghapus user
2. Mampu mengubah password user
3. Mampu melakukan koneksi ke Database Oracle
4. Mampu membedakan fungsi dan penggunaan SYSTEM dan OBJECT PRIVILEGES
5. Mampu menggunakan perintah GRANT dan REVOKE untuk memberikan dan mencabut SYSTEM dan OBJECT PRIVILEGES
6. Mampu menciptakan ROLE

#### Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 120 menit
2. Kegiatan Mandiri : 2 x 120 menit

#### Parameter Penilaian

1. Tugas Pendahuluan 40%
2. Jurnal : Hasil Pengamatan 20%
3. Tugas Akhir 40%

## 1.2 PERTANYAAN PENDAHULUAN

Kerjakan tugas pendahuluan ini, jika sudah selesai, kumpulkan kepada asisten anda. Waktu penggerjaan maksimal 10 menit.

1. Perintah apakah yang digunakan untuk menciptakan user ? Berikan contoh !
2. Jelaskan apa yang dimaksud dengan privileges !
3. Jelaskan perbedaan *system privileges* & *object privileges* !
4. Jelaskan apa yang dimaksud dengan role !
5. Perintah apakah yang diberikan untuk memberikan hak akses kepada seorang user ?
6. Perintah apakah yang diberikan untuk mencabut hak akses dari seorang user ?

Praktik

### 1.2.1 Menciptakan User

Pada bagian ini, akan dipelajari mengenai

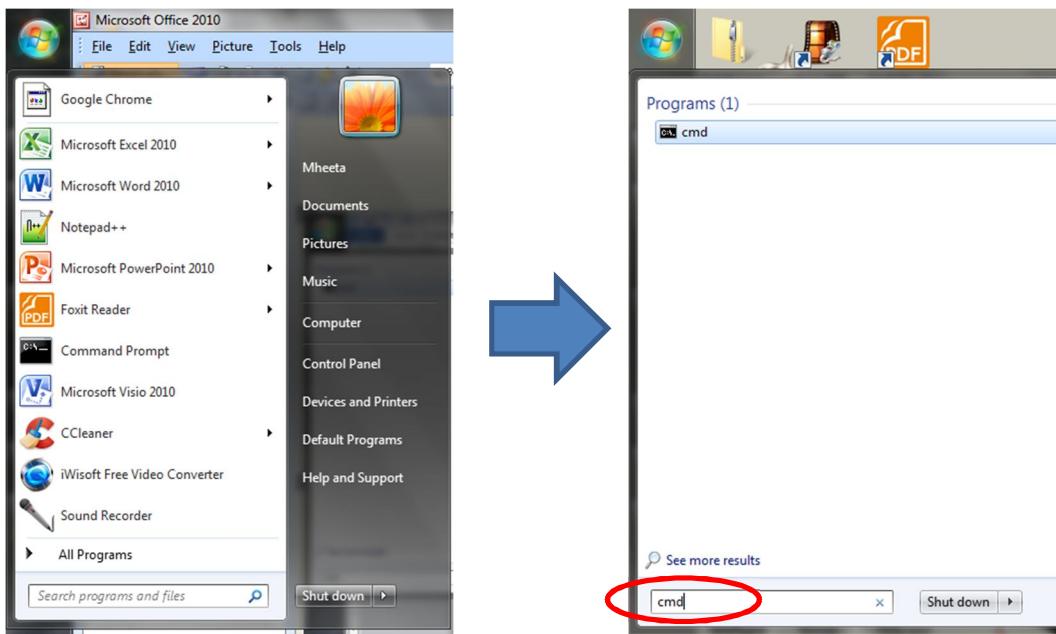
1. Cara menciptakan user,
2. Melakukan koneksi ke database Oracle

#### 1.2.1.1 Soal

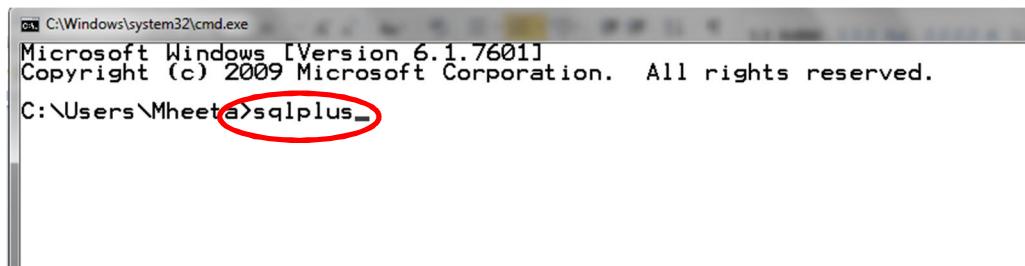
1. Ciptakan user dengan nama "sita" dan password "rabbit".
2. Setelah user "sita" selesai diciptakan, lakukan koneksi ke database Oracle.

#### 1.2.1.2 Langkah Penyelesaian

1. Ciptakan user dengan nama "sita" dan password "rabbit".
  - a. Masuk ke dalam sqlplus terlebih dahulu, dengan cara:  
Dari menu Start Windows, ketikkan "cmd" untuk masuk ke command prompt. Lalu tekan tombol "Enter"



Setelah masuk dalam mode command prompt, ketikkan "sqlplus" untuk masuk ke PL/SQL Oracle.



- b. Setelah itu masuk sebagai user "system" untuk dapat menciptakan user. Isikan *password* user "system" sesuai dengan *password* yang Anda isikan pada saat melakukan instalasi Oracle.

A screenshot of a Windows command prompt window titled 'C:\Windows\system32\cmd.exe - sqlplus'. The window displays the following text:  
 Microsoft Windows [Version 6.1.7601]  
 Copyright (c) 2009 Microsoft Corporation. All rights reserved.  
 C:\Users\Mheeta>sqlplus  
 SQL\*Plus: Release 11.2.0.2.0 Production on Fri Jan 17 21:02:30 2014  
 Copyright (c) 1982, 2010, Oracle. All rights reserved.  
 Enter user-name: system  
 Enter password: → Isikan dengan password yang diinputkan saat instalasi lalu tekan "ENTER"

- c. Untuk menciptakan user dalam Oracle adalah dengan menggunakan perintah CREATE USER. Adapun bentuk umum perintah CREATE USER adalah sebagai berikut:

```
CREATE USER <user>
IDENTIFIED BY <password>;
```

Untuk menciptakan user "sita" dengan password "rabbit", pada prompt SQL ketikkan:

```
SQL> CREATE USER sita
    IDENTIFIED BY rabbit;
```

```
SQL> CREATE USER sita
2  IDENTIFIED BY rabbit;
User created.
SQL> _
```

2. Setelah user "sita" selesai diciptakan, lakukan koneksi ke database Oracle.
- Untuk masuk atau koneksi sebagai user "sita", lakukan *disconnect* (atau *logout*) dari user "system" terlebih dahulu.

```
SQL> CREATE USER sita
2  IDENTIFIED BY rabbit;
User created.
SQL> DISCONNECT
Disconnected from Oracle Database 11g Express Edition Release 11.2.0.2.0 - Production
SQL> _
```

- Untuk melakukan koneksi ke database dan masuk ke dalam *schema* user "sita", maka Anda login sebagai user "sita" sebagai berikut:

```
SQL> conn sita
Enter password: Inputkan password dengan "rabbit"
ERROR:
ORA-01045: user SITA lacks CREATE SESSION privilege; logon denied
SQL> _
```

Perhatikan pesan kesalahan yang muncul:

**ERROR:**

**ORA-01045: user SITA lacks CREATE SESSION privilege; logon denied**



**PERTANYAAN !!**

1. Apakah arti pesan kesalahan tersebut ? Jelaskan jawaban Anda !
2. Ciptakan user baru dengan nama dan password sesuai dengan kelas Anda masing-masing. Misal: PIS1301, PIS1405, dst....
3. Ciptakan user Scott dengan password "scott".
4. Ciptakan user Alice dengan password "alice".

Praktekkan dan kemudian tuliskan jawaban Anda dalam Jurnal Praktikum Anda.

#### 1.2.1.3 Solusi Lengkap

1. Ciptakan user dengan nama "sita" dan password "rabbit".

```
SQL> CREATE USER sita  
      IDENTIFIED BY rabbit;  
  
SQL> DISCONNECT
```

2. Setelah user "sita" selesai diciptakan, lakukan koneksi ke database Oracle.

```
SQL> CONN sita;
```

atau

```
SQL> CONN sita/rabbit;
```

>Password user "sita"

### 1.2.2 Pengantar Privileges (Hak Akses)

Pada bagian ini, akan dipelajari mengenai

1. Memberikan hak akses pada user,
2. Mengubah password user,
3. Mencabut hak akses user

#### 1.2.2.1 Soal

1. Berikan hak akses kepada user "sita" agar dapat melakukan koneksi ke Oracle database.
2. Ubah password user "sita" menjadi "kelinci"
3. Cabut hak akses user "sita" untuk melakukan koneksi ke Oracle database
4. Buat table BARANG berikut ini dalam schema "sita", isikan 3 baris record dalam table tersebut. Berikan hak akses kepada user "<kelas Anda>" agar dapat melihat data pada tabel BARANG yang ada pada schema "sita"

**TABLE BARANG**

No	Nama Field	Tipe Data	Keterangan
1	bid	Char(4)	PK
2	bnama	Varchar2(25)	
3	warna	Varchar2(20)	

#### 1.2.2.2 Langkah Penyelesaian

1. Berikan hak akses kepada user "sita" agar dapat melakukan koneksi ke Oracle database.
  - a. Agar seorang user dapat melakukan koneksi ke Oracle database, maka user tersebut harus diberikan hak akses. Bentuk umum perintah untuk memberikan hak akses adalah:

```
GRANT <privilege1, privilege2,...>
TO <user1,user2,...|role|PUBLIC>;
```

- b. Untuk memberikan hak akses koneksi kepada user "sita" dengan menggunakan privilege "CONNECT" sebagai berikut:
  - Masuk sebagai user "system" (masuk ke dalam schema "system")
  - Dari prompt SQL ketikkan:

```
SQL> GRANT CONNECT TO sita;
```

```
SQL> GRANT CONNECT TO sita;
```

```
Grant succeeded.
```

```
SQL> -
```

- c. Kemudian keluar dari schema "system" dengan perintah "disconnect" dan masuk ke schema "sita" sebagai berikut:

```
SQL> disconnect
Disconnected from Oracle Database 11g Express Edition Release 11.2.0.2.0 - F
ction
SQL> conn sita/rabbit
Connected.
SQL>
```

2. Ubah password user "sita" menjadi "kelinci"

- a. Keluar dari schema "sita" dengan perintah "disconnect" dan masuk ke schema "system" sebagai berikut:

```
SQL> conn sita/rabbit
Connected.
SQL> disconnect
Disconnected from Oracle Database 11g Express Edition Release 11.2.0.2
ction
SQL> conn system
Enter password:
Connected.
SQL>
```

- b. Perintah untuk mengubah password user dengan menggunakan perintah ALTER USER sebagai berikut:

```
SQL> ALTER USER sita
          2 IDENTIFIED BY kelinci;
User altered.
SQL>
```

- c. Kemudian coba masuk ke schema "sita" dengan menggunakan password yang baru.

```
SQL> disconnect
Disconnected from Oracle Database 11g Express Edition
ction
SQL> conn sita/kelinci
Connected.
SQL>
```

3. Cabut hak akses user "sita" untuk melakkan koneksi ke Oracle Database
  - a. Bentuk umum perintah untuk mencabut hak akses seorang user adalah sebagai berikut:

```
REVOKE <privilege1, privilege2,... | [ALL]>
  ON object
  FROM <user1, user2,... | ROLE | PUBLIC>
  [CASCADE CONSTRAINTS];
```

- b. Perintah untuk memberikan hak akses koneksi ke database adalah dengan menggunakan privilege "CONNECT". Agar user "sita" tidak dapat melakukan koneksi ke database maka privilege "CONNECT" yang telah diberikan sebelumnya kepada user "sita" akan kita cabut dengan menggunakan perintah REVOKE.
    - Masuk ke dalam schema "system", lalu berikan perintah sebagai berikut pada prompt SQL.

```
SQL> REVOKE CONNECT FROM sita;
```

```
SQL> conn system
Enter password:
Connected.
SQL> REVOKE CONNECT FROM sita;
Revoke succeeded.
SQL>
```

- c. Untuk membuktikan perintah tersebut berhasil atau tidak, keluar dari schema "system" dan lakukan koneksi ke schema "sita"

```
SQL> disconnect
Disconnected from Oracle Database 11g Express Edition Release 11.2.0
SQL> conn sita/kelinci;
ERROR:
ORA-01045: user SITA lacks CREATE SESSION privilege; logon denied
SQL>
```

Perhatikan pesan kesalahan yang muncul:

```
ERROR:  
ORA-01045: user SITA lacks CREATE SESSION privilege; logon denied
```



Hal ini menandakan bahwa user "sita" **TIDAK** memiliki **HAK AKSES** untuk melakukan koneksi ke Oracle database.

#### **PERTANYAAN !!**

3. Kembalikan hak akses user "sita" agar dapat melakukan koneksi ke Oracle database
4. Berikan hak akses kepada user "<kelas Anda>", user Scott dan user Alice agar dapat melakukan koneksi ke Oracle Database ! Termasuk jenis hak akses apakah itu ?
5. Cabut hak akses user "<kelas Anda>" agar tidak dapat melakukan koneksi ke Oracle database.
6. Kembalikan lagi hak akses user "<kelas Anda>" agar dapat melakukan koneksi ke Oracle Database

Praktekkan dan kemudian tuliskan jawaban Anda dalam Jurnal Praktikum Anda.

4. Buat table BARANG sebagai berikut dalam schema "sita", isikan 3 baris record dalam table tersebut. Berikan hak akses kepada user "<kelas Anda>" agar dapat melihat data pada tabel BARANG yang ada pada schema "sita"

**TABLE BARANG**

No	Nama Field	Tipe Data	Keterangan
1	bid	Char(4)	PK
2	bnama	Varchar2(25)	
3	warna	Varchar2(20)	

- a. Agar user "sita" dapat menciptakan tabel barang dalam schemanya, maka user "sita" **HARUS** diberikan hak akses untuk diperbolehkan menciptakan tabel dalam schemanya sebagai berikut:

- Masuk ke dalam schema "system" dan ketikkan perintah berikut ini pada prompt SQL:

```
SQL> GRANT RESOURCE TO sita;
```

- Setelah itu keluar dari schema "system" dan masuk ke schema "sita"

```
SQL> CONN sita/kelinci
```

- b. Buat perintah untuk menciptakan tabel barang terlebih dahulu, lalu jalankan dalam schema "sita".

```
CREATE TABLE barang
( bid                  CHAR(4),
  bNama                VARCHAR2(25),
  warna                VARCHAR2(20),
  CONSTRAINT pk_bid PRIMARY KEY (bid));
```

- c. Tambahkan 3 record dalam tabel barang dengan data-data sebagai berikut:

bid	bNama	warna
BG01	Kursi CT1	Merah
BG02	Kursi GT2	Hijau
BG04	Meja LH	Biru

```
INSERT INTO barang VALUES ('BG01','Kursi CT1','Merah');
INSERT INTO barang VALUES ('BG02','Kursi GT2','Hijau');
INSERT INTO barang VALUES ('BG04','Meja LH','Biru');
```

- d. Keluar dari schema "sita" dan kemudian masuk ke schema "<kelas Anda>". Lalu berikan perintah berikut ini: (dalam contoh ini schema "<kelas Anda>" adalah "PIS9999")

```
SQL> conn pis9999/pis9999
Connected.
SQL> select * from sita.barang;
select * from sita.barang
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> -
```

Perhatikan pesan kesalahan yang muncul

```
ERROR at line 1:
ORA-00942: table or view does not exist
```

Pesan kesalahan tersebut muncul dikarenakan user "pis9999" **TIDAK** memiliki hak akses untuk melihat data ke tabel BARANG yang ada di schema "sita".

**CATATAN:**

Agar memudahkan dalam praktek *controlling user access*, Anda dapat membuka 2 buah *command line*. Pada "start Window" ketikkan "cmd" kembali dan masuk ke "sqlplus". Maka sekarang Anda dapat bekerja dengan menggunakan 2 *command line*. Pada 1 *command line*, Anda dapat bekerja dengan schema "sita" dan pada *command line* yang lainnya dengan schema "<kelas Anda>".

- e. Agar user "<kelas Anda>" dapat melihat data tabel barang, user "<kelas Anda>" **HARUS** memiliki **HAK AKSES** untuk dapat melihat data pada tabel barang. Yang dapat memberikan **HAK AKSES** kepada user "<kelas Anda>" adalah user "system" atau user "sita". Dalam hal ini, user "sita" yang akan memberikan **HAK AKSES** kepada user "<kelas Anda>" agar dapat melihat data tabel BARANG yang ada pada schema "sita". Berikut perintah yang diberikan:  
(dalam contoh ini, user "<kelas Anda>" adalah user "pis9999".

```
SQL> GRANT SELECT ON BARANG TO PIS9999;
```

```
SQL> show user
USER is "SITA"
SQL> GRANT SELECT ON barang TO PIS9999;
Grant succeeded.
SQL>
```

- f. Masuk ke dalam schema "PIS9999" lalu berikan perintah sebagai berikut:

```
SQL> show user
USER is "PIS9999"
SQL> SELECT * FROM sita.barang;
BID  BNAMA          WARNA
---  -----
BG01 Kursi CT1      Merah
BG02 Kursi GT2      Hijau
BG04 Meja LH        Biru
SQL> _
```

Perhatikan bahwa user "pis9999" dapat melihat tabel barang yang ada pada schema "sita".

**PERTANYAAN !!**

7. Berikan hak akses kepada user "alice" untuk dapat **melihat, menambahkan dan menghapus** data pada tabel BARANG yang ada pada schema "sita" ! Termasuk jenis hak akses apakah itu ?
8. Melalui schema "alice" tambahkan data pada tabel BARANG yang ada pada schema "sita" dengan data-data sebagai berikut:  
(BG05, meja YT, hijau) dan (BG06, lemari RT, coklat)
9. Kemudian dari schema "sita", lakukan perintah "SELECT" untuk melihat data-data yang telah ditambahkan sebelumnya. Apa yang terjadi ? Jelaskan jawaban Anda ! Kemudian jelaskan punya langkah selanjutnya yang harus dilakukan agar semua perubahan yang dilakukan dapat dilihat.
10. Hapus data barang BG05 dari schema "alice". Apa yang terjadi ? Jelaskan jawaban Anda !
11. Melalui schema "alice", berikan hak akses kepada user "scott" agar dapat melihat data pada tabel BARANG yang ada pada schema "sita". Apa yang terjadi ? Jelaskan jawaban Anda !

Praktekkan dan kemudian tuliskan jawaban Anda dalam Jurnal Praktikum Anda.

### 1.2.3 With Grant Option , With Admin Option dan Role

Pada bagian ini, akan dipelajari mengenai

1. Memberikan hak akses user dengan perintah **With Grant Option**
2. Memberikan hak akses user dengan perintah **With Admin Option**
3. Role

#### 1.2.3.1 Soal

1. Berikan hak akses kepada user Alice agar dapat melihat dan menambahkan data pada tabel BARANG yang ada di schema"sita". Tambahkan pula hak akses ke user Alice agar dapat memberikan hak akses yang sama kepada user lainnya.
2. Lakukan uji coba terhadap hak akses yang diberikan kepada Alice.
  - a. Buat query untuk melihat seluruh data barang dari schema "alice".
  - b. Tambahkan data berikut ini kedalam tabel BARANG dari schema "alice"  
(BG09, Meja Rapat YT05, Coklat Tua)
  - c. Berikan hak akses kepada user "scott" untuk dapat melihat tabel BARANG dari schema "alice". Kemudia uji coba dengan membuat query untuk melihat seluruh data barang dari schema "scott".
3. Berikan hak akses kepada user Scott agar dapat menciptakan tabel dalam schemanya. Tambahkan pula hak akses kepada user Scott agar dapat memberikan hak akses yang sama kepada user yang lainnya.

### 1.2.3.2 Langkah Penyelesaian

- Berikan hak akses kepada user Alice agar dapat **melihat dan menambahkan data** pada tabel BARANG yang ada di schema "sita". Tambahkan pula hak akses ke user Alice agar **dapat memberikan hak akses** yang sama kepada user lainnya.
  - Masuk kedalam schema "sita", lalu ketikkan perintah berikut ini pada prompt SQL.

```
SQL> show user
USER is "SITA"
SQL>
SQL>
SQL> GRANT select, insert ON barang TO alice WITH GRANT OPTION;
Grant succeeded.
SQL>
```

- Lakukan uji coba terhadap hak akses yang diberikan kepada Alice.
  - Buat query untuk melihat seluruh data barang dari schema "alice".

Dari schema "alice", ketikkan perintah berikut ini pada prompt SQL:

```
SQL> SELECT * FROM sita.BARANG;
```

- Tambahkan data berikut ini kedalam tabel BARANG dari schema "alice" (BG09, Meja Rapat YT05, Coklat Tua)

Dari schema "alice", ketikkan perintah berikut ini pada prompt SQL:

```
SQL> INSERT INTO sita.BARANG
      VALUES ('BG09','Meja Rapat YT05','Coklat Tua');

SQL> COMMIT;
```

- Berikan hak akses kepada user "scott" untuk dapat melihat tabel BARANG dari schema "alice". Kemudian uji coba dengan membuat query untuk melihat seluruh data barang dari schema "scott".

Dari schema "alice", ketikkan perintah berikut ini dari prompt SQL:

```
SQL> GRANT select ON sita.BARANG TO scott;
```

Untuk menguji bahwa user "scott" telah mendapat hak akses untuk melihat tabel BARANG, masuk ke schema "scott" lalu ketikkan perintah berikut ini pada prompt SQL:

```
SQL> SELECT * FROM sita.BARANG;
```

3. Berikan hak akses kepada user Scott agar dapat **menciptakan tabel** dalam schemanya. Tambahkan pula hak akses kepada user Scott agar **dapat memberikan hak akses** yang sama kepada user yang lainnya.
- Untuk memberikan *system privilege* kepada seorang user, dilakukan melalui schema "system". Masuk ke dalam schema "system", lalu ketikkan perintah berikut ini pada prompt SQL:

```
SQL> GRANT resource, create table TO scott  
      WITH ADMIN OPTION;
```

### **PERTANYAAN !!**

12. Lakukan pengujian terhadap hak akses yang diberikan kepada user "scott", dengan cara:
- a. Berikan hak akses yang sama (resource & create table) dari schema "scott" kepada user "<kelas Anda>".
  - b. Buat table BOATS pada schema "<kelas Anda>" dengan struktur sbb:  
bid              char(3)              PK  
bname            varchar2(20)  
color            varchar2(15)
  - c. Isikan data berikut dalam tabel BOATS tersebut:  
(101,Interlake, Blue) ; (102,Interlake, Red)
13. Hapus user "alice" melalui schema "system" dengan menggunakan perintah :  
DROP USER <namaUser> CASCADE;  
Kemudian masuk ke dalam schema "scott", buat query untuk melihat tabel BARANG. Apa yang terjadi ? Jelaskan jawaban Anda !

Praktekkan dan kemudian tuliskan jawaban Anda dalam Jurnal Praktikum Anda.

### 1.3 TEST AKHIR

1. Jelaskan dampak penggunaan sintak WITH GRANT OPTION serta WITH ADMIN OPTION saat memberikan dan mencabut *privileges* user !
2. Apabila Anda menciptakan tabel, siapakah yang dapat memberikan hak akses terhadap tabel yang sudah Anda ciptakan tersebut ?
3. Buat user dengan nama "jack" dan "brad" (password sama dengan nama usernya) !
4. Buatlah sebuah ROLE dengan nama MANAGER !
5. Berikan hak akses kepada ROLE MANAGER untuk dapat melakukan koneksi dalam database
6. Berikan hak akses ROLE MANAGER kepada user "jack" dan "brad"
7. Berikan hak akses pada ROLE MANAGER untuk dapat melihat, menambah dan menghapus data pada table BOATS yang ada pada schema "<kelas Anda>"
8. Dari schema "jack", tambahkan data 103, Marine, Yellow
9. Dari schema "brad", hapus data yang memiliki boat id 102
10. Hapus privilege untuk menambah dan menghapus data yang diberikan kepada user "jack" pada table BOATS.

#### 1.3.1 Lain-lain

1. Kumpulkan jawaban anda kepada para asisten sebelum praktikum diakhiri.
2. Media penggeraan dan pengumpulan bebas, ikuti aturan yang telah ditetapkan oleh asisten praktikum.

## 2 BAB II SCHEMA OBJECT

### 2.1 IDENTITAS

#### Kajian

Schema Object

#### Topik

1. Index
2. Synonym
3. Sequence

#### Referensi

1. Gavin Powell, C.M.D.(2005). Oracle SQL Jumpstart With Examples. USA: Elsevier Inc.
2. Greenberg, N. (Edition 1.1 August 2004). Oracle Database 10g: SQL Fundamental I. Jobi Varghese.

#### Kompetensi Utama

1. Mahasiswa dapat membuat index dan melakukan pemeliharaan pada index
2. Mahasiswa dapat membuat private dan public sysnonym
3. Mahasiswa dapat membuat sequence
4. Mahasiswa dapat melakukan modifikasi dan menghapus sequence

#### Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 120 menit
2. Kegiatan Mandiri : 2 x 120 menit

#### Parameter Penilaian

1. Tugas Awal
2. Jurnal Pengamatan
3. Tugas Akhir

## 2.2 PERTANYAAN PENDAHULUAN

Kerjakan tugas pendahuluan ini, jika sudah selesai, kumpulkan kepada asisten anda. Waktu penggeraan maksimal 15 menit.

1. Jelaskan apa yang dimaksud dengan *index* dan sebutkan 2 cara index dapat diciptakan !
2. Kapan index perlu untuk diciptakan ? Sebutkan minimal 3.
3. Kapan index tidak perlu untuk diciptakan ? Sebutkan minimal 3.
4. Jelaskan apa yang dimaksud dengan *synonym* !
5. Jelaskan apa yang dimaksud dengan *sequence* !

## Praktik

### 2.2.1 Index

Pada bagian ini, akan dipelajari mengenai penggunaan index dalam database.

#### 2.2.1.1 Soal

Untuk mengerjakan soal-soal berikut ini, sebelumnya:

- a. Pada schema HR, buat table EMP yang memiliki kolom-kolom emp\_id, dept\_no, fname, lname, emp\_job dan emp\_sal. Isikan data pada table EMP berdasarkan data pada table EMPLOYEES.
- b. Buatkan primary key pada table EMP. Primary key adalah **emp\_id**.

1. Buat index pada kolom lname yang ada pada table EMP. Beri nama index dengan **emp\_lname\_idx**
2. Periksa keberadaan index yang telah diciptakan pada soal no. 1
3. Hapus index emp\_lname\_idx

#### 2.2.1.2 Langkah Penyelesaian

##### Menciptakan table EMP:

```
CREATE TABLE emp(emp_id, dept_no, fname, lname, emp_job, emp_sal)
AS
    SELECT employee_id, department_id, first_name, last_name,
           job_id, salary
    FROM employees;
```

##### Membuat Primary Key:

```
ALTER TABLE emp
    ADD CONSTRAINT pk_emp_id PRIMARY KEY (emp_id);
```

1. Buat index pada kolom lname yang ada pada table EMP. Beri nama index dengan **emp\_lname\_idx**.

- Untuk membuat index pada satu atau lebih kolom, sintaks penulisannya:

```
CREATE INDEX index
ON table {column[,...column]...};
```

- Masuk ke dalam schema "HR". Dari prompt SQL ketikkan perintah berikut ini:

```
SQL> CREATE INDEX emp_lname_idx ON emp(lname);
```

2. Periksa keberadaan index yang telah diciptakan pada soal no. 1

- Untuk mengetahui keberadaan index, dapat dengan menggunakan data dictionary **USER\_INDEXES** dan **USER\_IND\_COLUMNS**

```
SQL> column column_name format a15
SQL> SELECT ic.index_name, ic.column_name,
          ic.column_position col_pos, ix.uniqueness
     FROM user_indexes ix, user_ind_columns ic
    WHERE ic.index_name = ix.index_name AND ic.table_name = 'EMP';
```

3. Hapus index emp\_lname\_idx

- Untuk menghapus index dari *data dictionary* digunakan perintah **DROP INDEX**
- Dari prompt SQL, ketikkan perintah berikut ini:

```
SQL> DROP INDEX emp_lname_idx;
```

### PERTANYAAN !!

1. Apa arti perintah "**column column\_name format a15**" pada contoh soal no. 2 diatas !
2. Buat sebuah nonunique index pada kolom DEPT\_NO di table EMP
3. Munculkan informasi tentang semua index yang ada pada table EMP.

Praktekkan dan kemudian tuliskan jawaban Anda dalam Jurnal Praktikum Anda.

## 2.2.2 Synonym

Pada bagian ini, akan dipelajari mengenai penggunaan synonym dalam database.

### 2.2.2.1 Soal

1. Buat sebuah synonym untuk table BOATS dengan nama "KAPAL"
2. Hapus synonym "KAPAL" untuk table BOATS

### 2.2.2.2 Langkah Penyelesaian

1. Buat sebuah synonym untuk table BOATS dengan nama "KAPAL"

- Sintaks untuk pembuatan synonym:

```
CREATE [PUBLIC] SYNONYM synonym
FOR object;
```

- Masuk dalam schema "<kelas Anda>", lalu ketikkan perintah berikut ini pada prompt SQL:

```
SQL> CREATE SYNONYM kapal
      FOR boats;
```

2. Hapus synonym "KAPAL" untuk table BOATS

- Sintaks untuk menghapus synonym

```
DROP SYNONYM nama_synonym;
```

- Pada prompt SQL, ketikkan perintah berikut ini:

```
SQL> DROP SYNONYM kapal;
```

### PERTANYAAN !!

1. Ciptakan sebuah PUBLIC synonym dengan nama "BRG" untuk table BARANG yang dimiliki oleh user "sita".
2. Hapus PUBLIC synonym "BRG"
3. Siapakah yang dapat menghapus sebuah PUBLIC synonym ?

Praktekkan dan kemudian tuliskan jawaban Anda dalam Jurnal Praktikum Anda.

### 2.2.3 Sequence

Pada bagian ini, akan dipelajari mengenai penggunaan sequence dalam database oracle.

#### 2.2.3.1 Soal

Untuk mengerjakan soal-soal berikut ini, sebelumnya:

- a. Pada schema HR, buat table DEPT yang memiliki kolom-kolom dept\_no, dname, loc. Isikan data pada table DEPT berdasarkan data pada table DEPARTMENTS.
- b. Buatkan primary key pada table DEPT. Primary key adalah **DEPT\_NO**.

1. Buat sequence DEPT\_DEPTNO yang akan digunakan sebagai PRIMARY KEY dalam table DEPT. Jangan gunakan pilihan CYCLE.
2. Periksa keberadaan dari sequence yang telah diciptakan
3. Gunakan sequence *dept\_deptno* untuk mengisi baris baru untuk department "MARKETING" yang berlokasi di "SAN DIEGO"
4. Buat query untuk melihat nilai saat ini dari sequence *dept\_deptno*
5. Ubah nilai maksimum dari sequence *dept\_deptno* menjadi 99999
6. Hapus sequence *dept\_deptno*

#### 2.2.3.2 Langkah Penyelesaian

1. Buat sequence DEPT\_DEPTNO yang akan digunakan sebagai PRIMARY KEY dalam table DEPT. Jangan gunakan pilihan CYCLE.
- Sintaks umum Sequence:

```
CREATE SEQUENCE sequence
[INCREMENT BY n]
[START WITH n]
[ {MAXVALUE n | NONMAXVALUE} ]
[ {MINVALUE n | NONMINVALUE} ]
[ {CYCLE | NONCYCLE} ]
[ {CACHE n | NOCACHE} ];
```

#### CATATAN:

- Yang diberi garis bawah adalah DEFAULT
- Pada prompt SQL, ketikkan perintah berikut ini:

```
SQL> CREATE SEQUENCE dept_deptno
INCREMENT BY 1
START WITH 91
MAXVALUE 100
NOCACHE
NOCYCLE;
```

2. Periksa keberadaan dari sequence yang telah diciptakan
  - Untuk memeriksa keberadaan dari sequence, informasi bisa diambil dari data dictionary USER\_SEQUENCES
  - Dari prompt SQL, ketikkan:

```
SQL> SELECT sequence_name, min_value, max_value,
       increment_by, last_number
  FROM user_sequences;
```
  
3. Gunakan sequence dept\_deptno untuk mengisi baris baru untuk department "MARKETING" yang berlokasi di "SAN DIEGO"
  - Untuk menerapkan penggunaan sequence dapat digunakan perintah NEXTVAL dan CURRVAL.
  - Next sequence ditempatkan pada NEXTVAL, sedangkan CURRVAL menyimpan *current sequence*
  - Dari prompt SQL, ketikkan:

```
SQL> INSERT INTO dept(dept_no, dname, loc)
   VALUES (dept_deptno.NEXTVAL, 'Marketing', 'San Diego');
```
  
4. Buat query untuk melihat nilai saat ini dari sequence dept\_deptno
  - Untuk melihat nilai dari sequence dept\_deptno saat ini:

```
SQL> SELECT dept_deptno.CURRVAL
   FROM DUAL;
```
  
5. Ubah nilai maksimum dari sequence dept\_deptno menjadi 99999
  - Perintah **ALTER SEQUENCE** *nama\_sequence* dapat digunakan untuk memodifikasi sequence, misal merubah increment value, maximum value, pilihan cycle, atau cache.
  - Ketikkan perintah berikut dari prompt SQL:

```
SQL> ALTER SEQUENCE dept_deptno
      INCREMENT BY 1
      MAXVALUE 99999
      NOCACHE
      NOCYCLE;
```
  
6. Hapus sequence dept\_deptno
  - Untuk menghapus sequence digunakan perintah:

```
DROP SEQUENCE nama_sequence
```

  - Sekali dihapus, sequence tidak bisa direferensi lagi.
  - Dari prompt SQL, ketikkan:

```
SQL> DROP SEQUENCE dept_deptno;
```

**PERTANYAAN !!**

1. Buat sequence untuk digunakan sebagai PRIMARY KEY pada table DEPARTMENTS. Sequence dimulai dari 60, dan mempunyai nilai maksimum 200. Sequence di-increment dengan angka 10. Nama sequence yang dibuat adalah DEPT\_ID\_SEQ.
2. Tampilkan keberadaan sequence dari dictionary USER\_SEQUENCES
3. Buat query untuk menambahkan dua record dalam table DEPARTMENTS. Gunakan sequence DEPT\_ID\_SEQ. Tambahkan dua departemen "Education" yang berlokasi di "Canada" dan departemen "Administration" yang berlokasi di "San Diego"

Praktekkan dan kemudian tuliskan jawaban Anda dalam Jurnal Praktikum Anda.

**2.3 TEST AKHIR**

1. Ciptakan synonym dengan nama "ship" untuk table BOATS
2. Buat sebuah nonunique index pada kolom bname pada table BOATS
3. Munculkan semua informasi tentang index yang ada pada table BOATS
4. Buat sequence untuk digunakan sebagai PRIMARY KEY pada table BOATS. Sequence dimulai dari 110, dan mempunyai nilai maksimum 300. Sequence di-increment dengan angka 5. Nama sequence yang dibuat adalah BOATS\_ID\_SEQ.
5. Modifikasi sequence DEPT\_DEPTNO, dimana sequence dimulai dari 100 dan mempunyai nilai maksimum 105. Sequence di-increment dengan angka 2. Gunakan CYCLE. Tambahkan 4 baris record (data terserah pada Anda). Amati apa yang terjadi, jelaskan jawaban Anda !!
6. Buat query untuk menambahkan dua buah record dalam table BOATS. Gunakan sequence BOATS\_ID\_SEQ. Tambahkan data kapal dengan nama "Flying" yang berwarna "Green" dan kapal dengan nama "Interlake" yang berwarna "Blue".

**2.3.1 Lain-lain**

1. Kumpulkan jawaban anda kepada para asisten sebelum praktikum diakhiri.
2. Media penggerjaan dan pengumpulan bebas, ikuti aturan yang telah ditetapkan oleh asisten praktikum.

### 3 BAB III VIEW

#### 3.1 IDENTITAS

##### Kajian

*Schema Object*

##### Topik

1. View

##### Referensi

1. Gavin Powell, C.M.D.(2005). Oracle SQL Jumpstart With Examples. USA: Elsevier Inc.
2. Greenberg, N. (Edition 1.1 August 2004). Oracle Database 10g: SQL Fundamental I. Jobi Varghese.

##### Kompetensi Utama

1. Mahasiswa memahami definisi view
2. Mahasiswa dapat membuat view
3. Mahasiswa dapat memanggil data melalui view
4. Mahasiswa dapat merubah definisi view
5. Mahasiswa dapat melakukan insert, update dan delete data melalui view
6. Mahasiswa dapat meghapus (drop) view

##### Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 120 menit
2. Kegiatan Mandiri : 2 x 120 menit

##### Parameter Penilaian

1. Pertanyaan Pendahuluan (Test Awal)
2. Jurnal Pengamatan
3. Test Akhir

### 3.2 PERTANYAAN PENDAHULUAN

Kerjakan tugas pendahuluan ini, jika sudah selesai, kumpulkan kepada asisten anda. Waktu penggeraan maksimal 15 menit.

1. Jelaskan apa yang dimaksud dengan view !
2. Sebutkan kegunaan dari view ! (4)
3. Jelaskan perbedaan simple views dan complex views !
4. Jelaskan tujuan penggunaan klausa WITH CHECK OPTION dalam view !

### 3.3 PRAKTIK

#### 3.3.1 View

Pada bagian ini dipelajari membuat view dalam database Oracle.

##### 3.3.1.1 Soal

###### GUNAKAN SCHEMA HR !

1. Buat view dengan nama EMPVU10 yang berisi detail dari pegawai (employee\_id, first\_name, last\_name, job\_id) yang bekerja pada departemen 10.
2. Tampilkan struktur view EMPVU10
3. Tampilkan semua data yang ada pada view EMPVU10
4. Modifikasi judul kolom dari EMPVU10 (soal no. 1) menjadi empno, ename, job. Catatan: ename adalah gabungan dari first\_name dan last\_name.
5. Hapus view EMPVU10.

##### 3.3.1.2 Langkah Penyelesaian

1. Buat view dengan nama EMPVU10 yang berisi detail dari pegawai (employee\_id, first\_name, last\_name, job\_id) yang bekerja pada departemen 10.

- View dapat dibuat dengan perintah **CREATE VIEW**.
- Dari prompt SQL, ketikkan perintah berikut ini:

```
SQL> CREATE VIEW EMPVU10
      AS SELECT employee_id, first_name, last_name, job_id
            FROM employees
           WHERE department_id = 10;
```

2. Tampilkan struktur view EMPVU10.

- Untuk menampilkan struktur dari view diberikan perintah **DESCRIBE namaview**.
- Dari prompt SQL, ketikkan:

```
SQL> DESCRIBE EMPVU10;
```

3. Tampilkan semua data yang ada pada view EMPVU10

- Dari prompt SQL, ketikkan:

```
SQL> SELECT * FROM EMPVU10;
```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	JOB_ID
200	Jennifer	Whalen	AD_ASST

4. Modifikasi judul kolom dari EMPVU10 (soal no. 1) menjadi empno, ename, job. Catatan: ename adalah gabungan dari first\_name dan last\_name.

- Untuk memodifikasi View digunakan klaus **CREATE OR REPLACE VIEW *namaView***;
- Dari prompt SQL, ketikkan:

```
SQL> CREATE OR REPLACE VIEW empvu10(empno, ename, job)
      AS SELECT employee_id, first_name||' '||last_name, job_id
            FROM employees
           WHERE department_id = 10;
```

- Untuk melihat data dalam view, ketikkan

```
SQL> SELECT * FROM EMPVU10;
```

EMPNO	ENAME	JOB
200	Jennifer Whalen	AD_ASST

5. Hapus view EMPVU10.

- View dapat dihapus dengan menggunakan perintah **DROP VIEW *namaView***;
- Dari prompt SQL, ketikkan:

```
SQL> DROP VIEW empvu10;
```

### 3.4 TEST AKHIR

1. Buat view SALVU30 (gunakan table EMP) yang berisi nomor, nama dan gaji pegawai yang bekerja di departemen 30. Beri nama kolom dengan EMPLOYEE\_NUMBER, NAME dan SAL\_EMP.  
**CATATAN:** NAME adalah gabungan dari FIRST\_NAME dan LAST\_NAME.
2. Ubah salary employee\_id 119 menjadi 2600 melalui view SALVU30.
3. Buat kompleks view DEPT\_SUM\_VU yang berisi nama departemen, minimum gaji, maksimum gaji, rata-rata gaji dari seluruh pegawai pada tiap-tiap departemen. Gunakan table EMPLOYEES.
4. Ubah gaji pegawai 206 menjadi 9500 melalui view DEPT\_SUM\_VU. Apa yang terjadi ? Jelaskan jawaban Anda !
5. Buat view EMPVU20 yang berisi semua data pegawai pada table EMPLOYEES yang bekerja di departemen 20. Beri klausa WITH CHECK OPTION.
6. Ubah data pegawai melalui view EMPVU20, dimana department\_id 10 diubah menjadi 7788. Apa yang terjadi ? Jelaskan jawaban Anda !
7. Buat view EMPVU10 yang berisi data nomor, nama dan pekerjaan pegawai (employee\_id, first\_name, last\_name, job\_id) untuk pegawai yang bekerja di departemen 10. Gunakan klausa READ ONLY .
8. Ubah data pegawai melalui view EMPVU10 dimana employee\_id 110 diubah menjadi 120. Apa yang terjadi ? Jelaskan jawaban Anda !
9. Buat view dengan nama EMPLOYEES\_VU berdasarkan pada nomor pegawai, nama pegawai (first name dan last name) serta nomor departemen dari table EMPLOYEES. Ubah kolom bagi nama pegawai menjadi PEGAWAI.
10. Munculkan seluruh data dari view EMPLOYEES\_VU
11. Munculkan nama view (view name) dan text dari data dictionary USER\_VIEWS
12. Dengan menggunakan view EMPLOYEES\_VU, buat query untuk menampilkan seluruh nama pegawai dan nomor departemen
13. Buat view dengan nama DEPT50 yang berisi nomor, nama (last name) dan gaji pegawai dari pegawai yang bekerja di departemen 50. Beri judul kolom dengan EMPLOYEE\_ID, EMPLOYEE dan DEPARTMENT\_ID. Jangan perbolehkan pegawai untuk mengisikan datanya lagi ke departemen lainnya melalui view.
14. Tampilkan struktur view DEPT50
15. Berikan perintah berikut ini pada view DEPT50:  
`UPDATE dept50 SET deptno = 80 WHERE employee = 'Matos';`  
Apa yang terjadi ? Jelaskan jawaban Anda !

## 4 BAB IV JOIN TABLE

### 4.1 IDENTITAS

#### Kajian

Query untuk manipulasi data dalam database

#### Topik

1. Inner Join
2. Left Outer Join
3. Right Outer Join
4. Full Outer Join

#### Referensi

1. Gavin Powell, C.M.D.(2005). Oracle SQL Jumpstart With Examples. USA: Elsevier Inc.
2. Greenberg, N. (Edition 1.1 August 2004). Oracle Database 10g: SQL Fundamental I. Jobi Varghese.

#### Kompetensi Utama

1. Mahasiswa dapat menulis statement SELECT yang mengakses data ke lebih dari satu table dengan menggunakan operator join (inner join, left join dan right join)
2. Mahasiswa dapat menampilkan data yang tidak memenuhi kondisi join dengan menggunakan operator outer join
3. Mahasiswa dapat melakukan join terhadap table itu sendiri (self join)

#### Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 120 menit
2. Kegiatan Mandiri : 2 x 120 menit

#### Parameter Penilaian

1. Jurnal Pengamatan
2. Tugas Akhir

## 4.2 PERTANYAAN PENDAHULUAN

Kerjakan tugas pendahuluan ini, jika sudah selesai, kumpulkan kepada asisten anda. Waktu penggeraan maksimal 15 menit.

1. Jelaskan perbedaan inner join, left join, right join serta full outer join !
2. Jelaskan apa yang disebut dengan self join !

## 4.3 PRAKTIK

### 4.3.1 Inner Join

Pada bagian ini akan dipelajari mengenai inner join dalam Oracle

#### 4.3.1.1 Soal

1. Tampilkan nama pegawai (last name) dan nama departemen tempat dimana pegawai tersebut bekerja. Gunakan schema "hr".
2. Tampilkan nomor pegawai, nama pegawai (last name), nama departemen serta kota departemen dari pegawai yang bernama "King". Gunakan schema "hr".

#### 4.3.1.2 Langkah Penyelesaian

1. Tampilkan nama pegawai (last name) dan nama departemen tempat dimana pegawai tersebut bekerja. Gunakan schema "hr".
  - a. Untuk dapat menyusun query dengan baik, langkah pertama yang harus dilakukan adalah memahami soal lalu menentukan table yang akan terlibat berdasarkan soal. Berdasarkan soal, yang dimunculkan adalah nama pegawai dan nama departemen. Nama pegawai ada pada table EMPLOYEES dan nama departemen ada pada table DEPARTMENTS
  - b. Berdasarkan poin a, terdapat 2 tabel yang terlibat. Perhatikan bahwa kedua table tersebut berelasi (lihat diagram schema "hr") berdasarkan kolom/field department\_id. Kolom tersebutlah yang akan dijoinkan dalam query yang akan dibuat.
  - c. Dari prompt SQL dalam schema "hr", ketikkan:

```
SELECT last_name, department_name
FROM employees e, departments d
WHERE e.department_id = d.department_id;
```

ATAU

```
SELECT last_name, department_name
FROM employees e JOIN departments d
ON e.department_id = d.department_id;
```

ATAU

```
SELECT last_name, department_name
FROM employees JOIN departments
USING (department_id);
```

Outputnya:

LAST_NAME	DEPARTMENT_NAME
Whalen	Administration
Fay	Marketing
Hartstein	Marketing
Tobias	Purchasing
Colmenares	Purchasing
Baida	Purchasing
Raphaely	Purchasing
Khoo	Purchasing
Himuro	Purchasing
Mavris	Human Resources
Feeney	Shipping
:	:
:	:

2. Tampilkan nomor pegawai, nama pegawai (last name), nama departemen serta kota departemen dari pegawai yang bernama "King". Gunakan schema "hr".
  - a. Dari soal, yang ingin dimunculkan adalah:
    - Nomor pegawai, yakni EMPLOYEE\_ID, ada pada table EMPLOYEES
    - Nama pegawai, yakni LAST\_NAME, ada pada table EMPLOYEES
    - Nama departemen, yakni DEPARTMENT\_NAME, ada pada table DEPARTMENTS
    - Kota departemen, yakni CITY, ada pada table LOCATIONS
    - Terdapat seleksi baris, dimana yang ingin dimunculkan hanya informasi untuk pegawai yang memiliki last\_name = "King"
  - b. Berdasarkan poin a, dapat dilihat bahwa query yang akan ciptakan melibatkan 3 buah table, yaitu table EMPLOYEES, DEPARTMENTS dan LOCATION.
  - c. Dari schema "hr" dapat dilihat bahwa table EMPLOYEES dan DEPARTMENTS berelasi melalui field/kolom DEPARTMENT\_ID. Sedangkan untuk table DEPARTMENTS dan LOCATIONS berelasi melalui field/kolom LOCATION\_ID. Dengan demikian, pada prompt SQL, ketikkan:

```
SELECT employee_id, last_name, department_name, city
FROM employees e, departments d, locations l
WHERE (e.department_id = d.department_id)
AND (d.location_id = l.location_id)
AND LOWER(last_name) = 'king';
```

ATAU

```
SELECT employee_id, last_name, department_name, city
  FROM employees e
  JOIN departments d ON (e.department_id = d.department_id)
  JOIN locations l ON (d.location_id = l.location_id)
 WHERE LOWER(last_name) = 'king';
```

ATAU

```
SELECT employee_id, last_name, department_name, city
  FROM employees JOIN departments USING (department_id)
  JOIN locations USING (location_id)
 WHERE LOWER(last_name) = 'king';
```

Outputnya:

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_NAME	CITY
100	King	Executive	Seattle
156	King	Sales	Oxford

SQL> ■

### PERTANYAAN !!

1. Munculkan semua nama pegawai (first\_name dan last\_name) yang mulai masuk bekerja setelah tahun 2007.
2. Munculkan pegawai-pegawai yang memiliki nama (last\_name) yang diakhiri dengan huruf "n". Buat 2 buah query yang berbeda untuk menyelesaikan masalah tersebut !
3. Munculkan jumlah pegawai yang ada di setiap departemen.
4. Munculkan nama pegawai (gabungan antara first name dan last name) serta gaji mereka, untuk pegawai yang memiliki gaji tidak dalam rentang 5000 dan 12000. Urutkan berdasarkan gaji dari kecil ke besar.
5. Munculkan nama pegawai (last\_name), pekerjaan (job\_id) serta gaji untuk pegawai yang jobnya adalah SA\_REP atau ST\_CLERK dan tidak memiliki gaji 2500, 3500 atau 7000.

Praktekkan dan kemudian tuliskan jawaban Anda dalam Jurnal Praktikum Anda.

### 4.3.2 Left Outer Join, Right Outer Join & Full Outer Join

Pada bagian ini akan dipelajari mengenai left outer join, right outer join serta full outer join dalam oracle

#### 4.3.2.1 Soal

1. Munculkan semua nama pegawai (gabungan first name serta last name) serta nama departemen.  
CATATAN: pegawai yang tidak ditempatkan dalam departemen tertentu juga dimunculkan datanya. Gunakan LEFT OUTER JOIN.
2. Soal yang sama dengan no. 1 tetapi menggunakan RIGHT OUTER JOIN.
3. Soal yang sama dengan no. 1 tetapi menggunakan FULL OUTER JOIN

#### 4.3.2.2 Langkah Penyelesaian

1. Munculkan semua nama pegawai (gabungan first name serta last name) serta nama departemen.  
CATATAN: pegawai yang tidak ditempatkan dalam departemen tertentu juga dimunculkan datanya. Gunakan LEFT OUTER JOIN.
  - a. Data yang dimunculkan adalah:
    - Gabungan first name dan last name. Data diambil dari table EMPLOYEES
    - Nama departemen. Data diambil dari table DEPARTMENTS.
    - Kedua table berelasi dengan menggunakan field/kolom department\_id.
  - b. Berdasarkan poin a, berikut sintaks SQLnya:

```
SELECT first_name||' '||last_name emp_name, department_name
FROM employees e LEFT OUTER JOIN departments d
ON (e.department_id = d.department_id);
```

2. Soal yang sama dengan no. 1 tetapi menggunakan RIGHT OUTER JOIN.
  - a. Tabel yang digunakan sama dengan pembahasan pada no. 1. Hanya untuk soal no. 2 menggunakan RIGHT JOIN.
  - b. Sintaks SQL lengkapnya:

```
SELECT first_name||' '||last_name emp_name, department_name
FROM employees e RIGHT OUTER JOIN departments d
ON (e.department_id = d.department_id);
```

3. Soal yang sama dengan no. 1 tetapi menggunakan FULL OUTER JOIN
- Tabel yang digunakan sama dengan pembahasan pada no. 1.
  - Sintaks SQL lengkapnya:

```
SELECT first_name || ' ' || last_name emp_name, department_name
FROM employees e FULL OUTER JOIN departments d
ON (e.department_id = d.department_id);
```



### **PERTANYAAN !!**

Cermati hasil output yang muncul dari tiga query berbeda diatas (penggunaan LEFT OUTER JOIN, RIGHT OUTER JOIN dan FULL OUTER JOIN). Dari sudut pandang data yang dimunculkan, jelaskan jawaban Anda apa yang berbeda dari tiga query tersebut !!

Praktekkan dan kemudian tuliskan jawaban Anda dalam Jurnal Praktikum Anda.

### **4.3.3 Self Join**

Pada bagian ini akan dipelajari mengenai self join dalam oracle

#### **4.3.3.1 Soal**

- Tampilkan nama manager dari pegawai yang bernama "Blake"

#### **4.3.3.2 Langkah Penyelesaian**

- Tampilkan nama manager dari pegawai yang bernama "Blake"
  - Seringkali sebuah table perlu dijoinkan dengan table itu sendiri. Berdasarkan soal, perlu dicari manager dari seorang pegawai dalam hal ini "Blake", maka table pegawai dijoinkan dengan table pegawai untuk mendapatkan nomor pegawai manager dan namanya.
  - Sintaks querynya adalah:

```
SELECT e.last_name PEGAWAI, m.last_name MANAGER
FROM employees e, employees m
WHERE e.manager_id = m.employee_id
AND LOWER(e.last_name) = 'blake' ;
```

#### 4.4 TEST AKHIR

1. Buat query untuk menampilkan nama pegawai, nomor departemen dan nama departemen dari semua pegawai.

LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
Whalen	10	Administration
Fay	20	Marketing
Hartstein	20	Marketing
Tobias	30	Purchasing
Colmenares	30	Purchasing
Baida	30	Purchasing
Raphaely	30	Purchasing
:		

2. Buat daftar unik dari semua pekerjaan pada departemen 30, tampilkan pula lokasi (kota) dari departemen 30 pada output.

JOB_ID	CITY
PU_MAN	Seattle
PU_CLERK	Seattle

3. Tampilkan nama pegawai, nama departemen dan lokasi dari semua pegawai yang memiliki komisi.

LAST_NAME	DEPARTMENT_NAME	CITY
Russell	Sales	Oxford
Partners	Sales	Oxford
Errazuriz	Sales	Oxford
Cambrault	Sales	Oxford
Zlotkey	Sales	Oxford
Tucker	Sales	Oxford
Bernstein	Sales	Oxford
:		

4. Tampilkan nama pegawai dan nama departemen untuk semua pegawai yang memiliki huruf 'A' pada namanya.

LAST_NAME	DEPARTMENT_NAME
Whalen	Administration
Fay	Marketing
Hartstein	Marketing
Baida	Purchasing
Raphaely	Purchasing
Tobias	Purchasing
:	

5. Buat query untuk menampilkan nama pegawai, pekerjaan, nomor departemen, dan nama departemen untuk semua pegawai yang bekerja di kota 'DALLAS'.

ENAME	JOB	DEPTNO	DNAME
Whalen	AD_ASST	10	Administration
Raphaely	PU_MAN	30	Purchasing
Khoo	PU_CLERK	30	Purchasing
Bajna	PU_CLERK	30	Purchasing
:			

6. Buat query untuk menampilkan nama pegawai dan nomor pegawai, nama manager dan nomor pegawai dari manager.

Employee	EMP#	Manager	Mgr#
Kumar	173	Cambrault	148
Bates	172	Cambrault	148
Smith	171	Cambrault	148
Fox	170	Cambrault	148
Bloom	169	Cambrault	148
Ozer	168	Cambrault	148
Hunold	103	De Haan	102
:			

7. Modifikasi query pada nomor 6, buat outer join untuk menampilkan pula data pegawai yang tidak mempunyai manager.

Employee	EMP#	Manager	Mgr#
Nayer	125	Weiss	120
Johnson	179	Zlotkey	149
Grant	178	Zlotkey	149
Livingston	177	Zlotkey	149
Taylor	176	Zlotkey	149
Hutton	175	Zlotkey	149
Abel	174	Zlotkey	149
King	100		
:			

8. Buat query yang menampilkan nama pegawai, nomor departemen, dan semua employee yang bekerja pada departemen yang sama dengan employee. Samakan judul kolom seperti pada hasil berikut:

DEPARTEMEN PEGAWAI	KOLEGA
80 Sully	Johnson
80 Sully	King
80 Sully	Kumar
80 Sully	Lee
80 Sully	Livingston
80 Sully	Marvins
:	

9. Buat query untuk menampilkan nama dan tanggal mulai bekerja dari pegawai yang tanggal bekerjanya setelah pegawai bernama 'URMAN'.

LAST_NAME	HIRE_DATE
Ernst	21-MAY-07
Lorentz	07-FEB-07
Popp	07-DEC-07
Himuro	15-NOV-06
Colmenares	10-AUG-07

:

10. Tampilkan semua nama pegawai dan tanggal kerjanya serta nama manager dan tanggal kerjanya dimana tanggal mulai kerja pegawai lebih dulu daripada tanggal mulai kerja managernya.

PEGAWAI	HIRE_DATE	MANAGER	HIRE_DATE
Kaufling	01-MAY-03	King	17-JUN-03
Raphaely	07-DEC-02	King	17-JUN-03
De Haan	13-JAN-01	King	17-JUN-03
Higgins	07-JUN-02	Kochhar	21-SEP-05
Baer	07-JUN-02	Kochhar	21-SEP-05
Mavris	07-JUN-02	Kochhar	21-SEP-05
Whalen	17-SEP-03	Kochhar	21-SEP-05
Greenberg	17-AUG-02	Kochhar	21-SEP-05

:

## 5 BAB V IN LINE VIEW

### 5.1 IDENTITAS

#### Kajian

Query untuk manipulasi data dalam database

#### Topik

1. In Line View

#### Referensi

1. Gavin Powell, C.M.D.(2005). Oracle SQL Jumpstart With Examples. USA: Elsevier Inc.
2. Greenberg, N. (Edition 1.1 August 2004). Oracle Database 10g: SQL Fundamental I. Jobi Varghese.

#### Kompetensi Utama

1. Mahasiswa mengetahui apa yang disebut dengan In Line View,
2. Mahasiswa mampu menerapkan penggunaan In Line View.

#### Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 120 menit
2. Kegiatan Mandiri : 2 x 120 menit

#### Parameter Penilaian

1. Jurnal Pengamatan
2. Tugas Akhir

## 5.2 PERTANYAAN PENDAHULUAN

1. Apa yang dimaksud dengan In Line View ?
2. Apa keuntungan menggunakan In Line View dibandingkan dengan object database View ?

## 5.3 PRAKTIK

### 5.3.1 Soal I

1. Buat query dengan menggunakan In Line View untuk menghitung jumlah pegawai di setiap departemen. Urutkan berdasarkan jumlah pegawai secara descending.
2. Modifikasi query soal no.1, dimana informasi yang akan dimunculkan hanya departemen yang memiliki jumlah pegawai diatas 10.
3. Buat query untuk menampilkan 5 pegawai (last\_name) yang mendapatkan gaji paling besar dari table EMPLOYEES.

### 5.3.2 Langkah Penyelesaian

1. Buat query dengan menggunakan In Line View untuk menghitung jumlah pegawai di setiap departemen.
  - a. Prinsip In Line View pada dasarnya adalah meletakkan sintaks query dalam klausus FROM dimana query tersebut akan dianggap dan berfungsi sebagai table.
  - b. Sintaks querinya adalah:

```
SELECT * FROM (SELECT department_id, count(employee_id)
jumlah
FROM employees GROUP BY department_id) xy
ORDER BY 2 DESC;
```

```
SQL> SELECT * FROM (SELECT department_id, count(employee_id) jumlah
2          FROM employees GROUP BY department_id) xy
3  ORDER BY 2 DESC;
DEPARTMENT_ID      JUMLAH
-----
      50            45
      80            34
     100             6
      30             6
      60             5
      90             3
      20             2
     110             2
      40             1
      10             1
                  1
DEPARTMENT_ID      JUMLAH
-----
      70             1
12 rows selected.
```

2. Modifikasi query soal no.1, dimana informasi yang akan dimunculkan hanya departemen yang memiliki jumlah pegawai diatas 10.
- Pembahasan sama seperti soal no. 1, hanya ditambahkan klaus WHERE.
  - Sintaks querynya adalah:

```
SELECT * FROM (SELECT department_id, count(employee_id)
jumlah
      FROM employees GROUP BY department_id) xy
WHERE jumlah > 10 ORDER BY jumlah DESC;
```

```
SQL> SELECT * FROM (SELECT department_id, count(employee_id) jumlah
2          FROM employees GROUP BY department_id) xy
3 WHERE jumlah > 10 ORDER BY jumlah DESC;
DEPARTMENT_ID      JUMLAH
-----  -----
      50          45
      80          34
```

3. Buat query untuk menampilkan 5 pegawai (last\_name) yang mendapatkan gaji paling besar dari table EMPLOYEES.
- Untuk memunculkan informasi tersebut, kita dapat membuat sebuah In Line View dimana In Line View yang dibuat merupakan query untuk menampilkan last\_name dan gaji pegawai yang diurutkan secara menurun (*descending*). Kemudian akan diambil 5 terbesar.
  - Sintaks querynya adalah:

```
SELECT ROWNUM as RANK, last_name, salary
  FROM (SELECT last_name, salary FROM employees
        ORDER BY salary DESC)
 WHERE ROWNUM <= 5;
```

```
SQL> SELECT ROWNUM as RANK, last_name, salary
2   FROM (SELECT last_name, salary FROM employees
3         ORDER BY salary DESC)
4 WHERE ROWNUM <= 5;
RANK LAST_NAME           SALARY
-----  -----
      1 King                24000
      2 Kochhar              17000
      3 De Haan              17000
      4 Russell               14000
      5 Partners              13500
```

#### 5.4 TEST AKHIR

1. Gunakan In Line View untuk menampilkan nama (last name), salary, maksimum salary serta selisih maksimum salary dengan salary setiap pegawai untuk semua pegawai yang ada di departemen 20.

LAST_NAME	SALARY	MAX_SAL	SELISIH
Hartstein	13000	13000	0
Fay	6000	13000	7000

2. Gunakan In Line View untuk menampilkan 2 nama departemen yang paling banyak memiliki pegawai serta jumlah pegawainya.

DEPARTMENT_NAME	JML_PEG
Shipping	45
Sales	34

3. Gunakan In Line View untuk menampilkan nama (last name) pegawai, kode departemen serta gaji pegawai untuk setiap pegawai yang gajinya diatas rata-rata dari rata-rata departemen mereka.

LAST_NAME	DEPARTMENT_ID	SALARY	RATA2
King	90	24000	19333.33
Hunold	60	9000	5760.00
Ernst	60	6000	5760.00
Greenberg	100	12008	8601.33
Faviet	100	9000	8601.33
Raphaely	30	11000	4150.00
:			

4. Gunakan In Line View untuk menampilkan 5 nama pegawai yang mendapatkan total komisi tahunan paling rendah untuk semua pegawai yang mendapatkan komisi (last\_name, salary, total bonus)

LAST_NAME	SALARY	TOTAL_BONUS
Marvins	7200	1.2
Kumar	6100	1.2
Banda	6200	1.2
Ande	6400	1.2
Lee	6800	1.2

5. Gunakan In Line View untuk menampilkan nama pegawai (last\_name), salary, kode departemen serta gaji tertinggi untuk setiap departemen dimana pegawai yang dimunculkan adalah yang memiliki gaji lebih rendah dari gaji tertinggi dalam departemen tersebut.

LAST_NAME	SALARY	DEPARTMENT_ID	MAXSAL
Faviet	9000	100	12008
Chen	8200	100	12008
Sciarrra	7700	100	12008
Urman	7800	100	12008
Popp	6900	100	12008
Khoo	3100	30	11000
Baida	2900	30	11000

:

## 6 BAB VI MERGE

### 6.1 IDENTITAS

#### Kajian

Manipulasi data dalam database

#### Topik

1. Merge

#### Referensi

1. Gavin Powell, C.M.D.(2005). Oracle SQL Jumpstart With Examples. USA: Elsevier Inc.
2. Greenberg, N. (Edition 1.1 August 2004). Oracle Database 10g: SQL Fundamental I. Jobi Varghese.

#### Kompetensi Utama

1. Mahasiswa memahami manfaat dan fungsi statement MERGE dalam query,
2. Mahasiswa mampu menerapkan penggunaan statement MERGE.

#### Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 120 menit
2. Kegiatan Mandiri : 2 x 120 menit

#### Parameter Penilaian

1. Jurnal Pengamatan
2. Tugas Akhir

## 6.2 PERTANYAAN PENDAHULUAN

1. Jelaskan fungsi dan kegunaan dari statement MERGE !
2. Kapan statement MERGE bisa digunakan dalam query yang akan dibangun ?

## 6.3 PRAKTIK

### 6.3.1 Soal I

1. Akan dilakukan sinkronisasi data...

#### 6.3.1.1 *Langkah Penyelesaian*

#### 6.4 TEST AKHIR

Pada 3 (tiga) buah soal dibawah ini, kerjakan setiap soal dengan menggunakan jenis perulangan WHILE, FOR dan REPEAT.

1. Buatlah sebuah aplikasi yang dapat digunakan untuk menampilkan dan menghitung deret kuadrat dari inputan yang diberikan oleh pengguna. Contoh
  - a. Input : 5  
Output :  $1 + 4 + 9 + 16 + 25 = 55$
  - b. Input : 10  
Output :  $1 + 4 + 9 + 16 + 25 + 36 + 49 + 64 + 81 + 100 = 385$
  - c. Input : 0  
Output : inputan tidak valid
  - d. Input : -6  
Output : Inputan tidak valid
2. Buatlah sebuah aplikasi yang dapat digunakan untuk menampilkan dan menghitung deret divergen dari inputan yang diberikan oleh pengguna. Contoh
  - a. Input : 6  
Output :  $1 - 2 + 3 - 4 + 5 - 6 = -3$
  - b. Input : 10  
Output :  $1 - 2 + 3 - 4 + 5 - 6 + 7 - 8 + 9 - 10 = -5$
  - c. Input : 11  
Output :  $1 - 2 + 3 - 4 + 5 - 6 + 7 - 8 + 9 - 10 = 6$
  - d. Input : 0  
Output : inputan tidak valid
  - e. Input : -6  
Output : Inputan tidak valid
3. Buatlah sebuah aplikasi yang dapat digunakan untuk menampilkan dan menghitung nilai factorial dari inputan yang diberikan oleh pengguna. Contoh
  - a. Input : 6  
Output :  $6! = 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 120$
  - b. Input : 10  
Output :  $10! = 10 \times 9 \times 8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 3628800$
  - c. Input : 0  
Output : inputan tidak valid
  - d. Input : -6  
Output : Inputan tidak valid

## 7 BAB VII OPERATOR SET

### 7.1 IDENTITAS

#### Kajian

Menggunakan query operator SET.

#### Topik

1. Operator SET

#### Referensi

1. Gavin Powell, C.M.D.(2005). Oracle SQL Jumpstart With Examples. USA: Elsevier Inc.
2. Greenberg, N. (Edition 1.1 August 2004). Oracle Database 10g: SQL Fundamental II. Jobi Varghese.

#### Kompetensi Utama

1. Mahasiswa memahami operator SET
2. Mahasiswa dapat menggunakan operator SET yang dikombinasikan dengan multiple query ke dalam single query.
3. Mahasiswa mampu mengontrol urutan dari baris yang dikembalikan

#### Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 120 menit
2. Kegiatan Mandiri : 2 x 120 menit

#### Parameter Penilaian

1. Jurnal Pengamatan
2. Tugas Akhir

## 7.2 PERTANYAAN PENDAHULUAN

1. Sebutkan dan jelaskan operator SET yang dapat digunakan dalam Oracle !
2. Dari setiap jenis operator SET yang ada, berikan contohnya masing-masing !
3. Apa perbedaan operator UNION dengan operator UNION ALL !

## 7.3 PRAKTIK

### 7.3.1 Soal

1. Tampilkan detail job saat ini dan job sebelumnya dari semua employee. Tampilkan tiap employee hanya sekali. Gunakan table EMPLOYEES dan JOB\_HISTORY.
2. Tampilkan nomor pegawai dan id job pegawai yang saat ini memiliki jabatan yang sama dengan yang mereka pegang sebelum memulai masa jabatan mereka dengan perusahaan.
3. Tampilkan nomor pegawai yang merubah pekerjaannya sedikitnya sekali.

#### 7.3.1.1 Langkah Penyelesaian

1. Tampilkan detail job saat ini dan job sebelumnya dari semua employee. Tampilkan tiap employee hanya sekali. Gunakan table EMPLOYEES dan JOB\_HISTORY.
  - Untuk menyelesaikan permasalahan tersebut diatas, dapat digunakan operator UNION.
  - Operator UNION dapat digunakan untuk mengembalikan kedua nilai dari table dengan mengeliminasi duplikasi yang ada.
  - Sintaks querynya adalah:

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

2. Tampilkan nomor pegawai dan id job pegawai yang saat ini memiliki jabatan yang sama dengan yang mereka pegang sebelum memulai masa jabatan mereka dengan perusahaan.
  - Untuk menyelesaikan soal diatas, dapat digunakan operator INTERSECT
  - Terlebih dahulu harus dicari pekerjaan pegawai saat ini serta history pekerjaan pegawai.
  - Dengan menggunakan operator INTERSECT, maka didapat pegawai yang memiliki jabatan yang sama dengan yang mereka pegang sebelum memulai masa jabatan mereka saat ini (saling beririsan)
  - Sintaks SQLnya adalah:

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

3. Tampilkan nomor pegawai yang pekerjaannya tidak pernah berubah
- Untuk menyelesaikan masalah tersebut, nomor pegawai serta job id pegawai dalam table JOB\_HISTORY dikurangi dari nomor pegawai serta job id pegawai yang ada pada table EMPLOYEES.
  - Hasil yang didapatkan adalah baris-baris yang ada dalam table EMPLOYEES tetapi tidak ada dalam table JOB\_HISTORY. Data tersebut merupakan data pegawai yang pekerjaannya belum berubah sama sekali.
  - Struktur querynya adalah:

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

#### 7.4 TEST AKHIR

1. Dengan menggunakan operator SET, munculkan nomor departemen dan nama departemen bagi departemen-departemen yang tidak memiliki job id ST\_CLERK.

DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
30	Purchasing
40	Human Resources
60	IT
70	Public Relations

:

2. Dengan menggunakan operator SET, munculkan id country dan nama country yang tidak memiliki departemen yang berlokasi di negara tersebut.

CO	COUNTRY_NAME
AR	Argentina
BE	Belgium
DK	Denmark
EG	Egypt
FR	France
IL	Israel
KW	Kuwait
ML	Malaysia
NG	Nigeria
ZM	Zambia
ZW	Zimbabwe

:

3. Dengan menggunakan operator SET, munculkan nomor dan nama pegawai untuk pegawai-pegawai yang dihired tahun 2007 dan memiliki gaji antara 3000-3500.

EMPLOYEE_ID	LAST_NAME
187	Cabrio

4. Dengan menggunakan operator SET, munculkan nama pegawai serta job id pegawai untuk jenis pekerjaan yang ada di departemen 30 tetapi tidak di departemen 80

LAST_NAME	JOB_ID
Baida	PU_CLERK
Colmenares	PU_CLERK
Himuro	PU_CLERK
Khoo	PU_CLERK
Raphaely	PU_MAN
Tobias	PU_CLERK

5. Dengan menggunakan operator SET, munculkan nomor, nama pegawai serta tanggal mulai bekerja yang mulai bekerja pada tahun 2007 serta nomor, nama pegawai serta tanggal mulai bekerja yang mulai bekerja pada tahun 2006.

EMPLOYEE_ID	LAST_NAME	HIRE_DATE
103	Hunold	03-JAN-06
104	Ernst	21-MAY-07
106	Pataballa	05-FEB-06
107	Lorentz	07-FEB-07
112	Urman	07-MAR-06
113	Popp	07-DEC-07
118	Himuro	15-NOV-06
119	Colmenares	10-AUG-07
124	Mourgos	16-NOV-07
126	Mikkilineni	28-SEP-06
127	Landry	14-JAN-07

:

## 8 BAB VIII SINGLE ROW SUBQUERY

### 8.1 IDENTITAS

#### Kajian

Mengerjakan SubQuery

#### Topik

1. Single Row Subquery
2. Penggunaan Group Function dalam Subqueries
3. Penggunaan subqueries dalam klausa HAVING

#### Referensi

1. Gavin Powell, C.M.D.(2005). Oracle SQL Jumpstart With Examples. USA: Elsevier Inc.
2. Greenberg, N. (Edition 1.1 August 2004). Oracle Database 10g: SQL Fundamental I. Jobi Varghese.

#### Kompetensi Utama

1. Mahasiswa mampu menggambarkan tipe persoalan yang dapat dipecahkan oleh subquery
2. Mahasiswa mampu mendefinisikan subquery
3. Mahasiswa mampu menulis single row subquery
4. Mahasiswa mampu menggunakan group function dalam subquery
5. Mahasiswa mampu menggunakan subquery dalam klausa Having

#### Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 120 menit
2. Kegiatan Mandiri : 2 x 120 menit

#### Parameter Penilaian

1. Jurnal Pengamatan
2. Tugas Akhir

## 8.2 PERTANYAAN PENDAHULUAN

1. Jelaskan apa yang disebut dengan Single Row Subquery !
2. Dalam klausa mana sajakah subquery dapat diletakkan ?
3. Sebutkan tipe dari subquery dan jelaskan masing-masing perbedaannya !

## 8.3 PRAKTIK

### 8.3.1 Soal 1

1. Buat query untuk menampilkan gaji pegawai yang lebih besar dari gaji yang dimiliki oleh pegawai bernama 'DAVIES'
2. Buat query untuk menampilkan nama pegawai, pekerjaan dan gaji pegawai yang memiliki gaji paling kecil
3. Buat query untuk menampilkan gaji terendah dari setiap departemen yang gajinya diatas gaji terendah dari departemen 50.

#### 8.3.1.1 Langkah Penyelesaian

1. Buat query untuk menampilkan nama pegawai yang memiliki gaji lebih besar dari gaji yang dimiliki oleh pegawai bernama 'DAVIES'
  - Untuk memecahkan persoalan tersebut, dibutuhkan dua query. Satu query untuk mencari gaji yang dimiliki oleh 'DAVIES' (merupakan inner query) dan query lain untuk mencari pegawai yang memiliki gaji lebih besar daripada gaji 'DAVIES' (merupakan outer query).
  - Inner query atau subquery akan menghasilkan suatu nilai yang nantinya dipakai oleh outer query atau main query.
  - Sintaks querynya adalah:

```
SELECT last_name FROM employees           3100
WHERE salary > 
      (SELECT salary FROM employees
       WHERE LOWER(last_name) = 'davies');
```

LAST_NAME	SALARY
King	24000
Kochhar	17000
De Haan	17000
Hunold	9000
Ernst	6000
Austin	4800
Pataballa	4800
Lorentz	4200
Greenberg	12008

:

2. Buat query untuk menampilkan nama pegawai, pekerjaan dan gaji pegawai yang memiliki gaji paling kecil.

- Untuk memecahkan persoalan tersebut, dibutuhkan dua query. Satu query untuk mencari gaji terendah dengan menggunakan fungsi agregat MIN (merupakan inner query) dan query lain untuk mencari pegawai yang memiliki gaji sama dengan gaji terendah (merupakan outer query).
- Sintaks querynya adalah:

```
SELECT last_name, job_id, salary
FROM employees
WHERE salary = 
      (SELECT MIN(salary)
       FROM employees);
```

2500

LAST_NAME	JOB_ID	SALARY
Olson	ST_CLERK	2100

3. Buat query untuk menampilkan gaji terendah dari setiap departemen yang diatas gaji terendah dari departemen 50.

- Untuk memecahkan persoalan tersebut, dibutuhkan dua query. Satu query untuk mencari gaji terendah dari departemen 50 (merupakan inner query) dan query lain untuk mencari gaji terendah untuk setiap departemen (merupakan outer query).
- Untuk mencari gaji terendah dari setiap departemen dapat digunakan klausula GROUP BY.
- Klausula HAVING digunakan untuk mencari gaji terendah dari setiap departemen yang nilainya diatas gaji terendah dari departemen 50.
- Sintaks querynya adalah:

```
SELECT department_id, MIN(salary)
FROM employees
GROUP BY department_id
HAVING MIN(salary) > 
      (SELECT MIN(salary)
       FROM employees
       WHERE department_id = 50);
```

2500

DEPARTMENT_ID	MIN(SALARY)
100	6900
30	2500
	7000
90	17000
20	6000
70	10000
110	8300
80	6100
40	6500
60	4200
10	4400

**PERHATIKAN QUERY BERIKUT INI !**

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

Praktekkan query tersebut, apa output yang dihasilkan, jelaskan jawaban Anda !

**8.4 TEST AKHIR**

- Buat query untuk menampilkan nomor pegawai, nama pegawai (first name) dan gaji dari seluruh pegawai yang memiliki gaji di atas rata-rata gaji pegawai. Urutkan berdasarkan gaji secara menaik.

EMPLOYEE_ID	FIRST_NAME	SALARY
203	Susan	6500
123	Shanta	6500
165	David	6800
113	Luis	6900
155	Oliver	7000
161	Sarath	7000
178	Kimberely	7000
164	Mattea	7200
172	Elizabeth	7300
171	William	7400
154	Nanette	7500

:

2. Buat query untuk menampilkan last name dan salary pegawai untuk setiap pegawai yang melapor (atau merupakan bawahan) dari Greenberg.

LAST_NAME	SALARY
Faviet	9000
Chen	8200
Sciarra	7700
Urman	7800
Popp	6900

3. Buat query untuk menampilkan nama, pekerjaan dan salary pegawai yang sama dengan pegawai 141 dan memiliki gaji yang lebih besar dari pegawai 143. Urutkan menaik berdasarkan salary.

LAST_NAME	JOB_ID	SALARY
Seo	ST_CLERK	2700
Mikkilineni	ST_CLERK	2700
Atkinson	ST_CLERK	2800
Rogers	ST_CLERK	2900
Davies	ST_CLERK	3100
Stiles	ST_CLERK	3200
Nayer	ST_CLERK	3200
Mallin	ST_CLERK	3300
Bissot	ST_CLERK	3300
Rajs	ST_CLERK	3500
Ladwig	ST_CLERK	3600

4. Buat query untuk menampilkan nama departemen dan rata-rata gaji dari setiap departemen untuk departemen-departemen yang rata-rata gajinya diatas dari rata-rata gaji seluruh pegawai. Urutkan secara menurun berdasarkan rata-rata gaji setiap departemen.

DEPARTMENT_NAME	RATA2
Executive	19333.3333
Accounting	10154
Public Relations	10000
Marketing	9500
Sales	8955.88235
Finance	8601.33333
Human Resources	6500

5. Buat query untuk menampilkan gaji tertinggi dari setiap job (nama dan id job) yang ada, dimana yang ditampilkan adalah gaji tertinggi yang nilainya dibawah dari gaji tertinggi yang ada didepartemen 80.

JOB_ID	JOB_TITLE	TERTINGGI
AD_ASST	Administration Assistant	4400
IT_PROG	Programmer	9000
MK_MAN	Marketing Manager	13000
SA REP	Sales Representative	11500
AC_MGR	Accounting Manager	12008
AC_ACCOUNT	Public Accountant	8300
FI_MGR	Finance Manager	12008
PU MAN	Purchasing Manager	11000
SH_CLERK	Shipping Clerk	4200
FI_ACCOUNT	Accountant	9000
MK REP	Marketing Representative	6000

:

## 9 BAB IX MULTIPLE Row & MULTIPLE COLUMN SUBQUERY

### 9.1 IDENTITAS

#### Kajian

Mengerjakan Subquery

#### Topik

1. Multiple Row Subquery
2. Menggunakan operator IN, ANY, ALL dalam multiple row subquery
3. Multiple Column Subquery
4. Column comparison dalam multiple column subqueries (pairwise & nonpairwise)

#### Referensi

1. Gavin Powell, C.M.D.(2005). Oracle SQL Jumpstart With Examples. USA: Elsevier Inc.
2. Greenberg, N. (Edition 1.1 August 2004). Oracle Database 10g: SQL Fundamental II. Jobi Varghese.

#### Kompetensi Utama

1. Mahasiswa mampu menggambarkan tipe persoalan yang dapat dipecahkan dengan menggunakan multiple row subquery
2. Mahasiswa mampu menggambarkan tipe persoalan yang dapat dipecahkan dengan menggunakan multiple column subquery
3. Mahasiswa mampu menulis multiple row subquery
4. Mahasiswa mampu menulis multiple column subquery
5. Mahasiswa mampu menggunakan operator IN, ANY, ALL dalam multiple row subquery

#### Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 120 menit
2. Kegiatan Mandiri : 2 x 120 menit

#### Parameter Penilaian

1. Jurnal Pengamatan
2. Tugas Akhir

## 9.2 PERTANYAAN PENDAHULUAN

1. Apa yang dimaksud dengan multiple row subquery, jelaskan !
2. Apa perbedaan penggunaan operator IN, ALL, dan ANY dalam multiple row subquery, jelaskan !
3. Apa yang dimaksud dengan multiple column subquery, jelaskan !

## 9.3 PRAKTIK

### 9.3.1 Soal

1. Munculkan nama, gaji dan nomor departemen pegawai yang memiliki gaji sama dengan gaji minimum di setiap departemen.
2. Munculkan pegawai yang bukan seorang IT Programmer (IT\_PROG) dan gajinya lebih kecil dari setiap IT programmer yang ada.
3. Munculkan nomor pegawai, nomor manager, nomor departemen untuk pegawai-pegawai yang memiliki manager dan bekerja dalam departemen yang sama dengan pegawai 178 dan 174. Kerjakan dengan menggunakan *Pairwise Comparison Subquery*.
4. Soal sama dengan no.3, hanya kerjakan dengan menggunakan Nonpairwise Comparison Subquery.

#### 9.3.1.1 Langkah Penyelesaian

1. Munculkan nama, gaji dan nomor departemen pegawai yang memiliki gaji sama dengan gaji minimum di setiap departemen.
  - Untuk menyelesaikan soal diatas, dapat dibuat 2 buah query.
  - Query pertama yang merupakan inner query, dibuat untuk menampilkan gaji minimum dari setiap departemen yang ada.
  - Query kedua yang merupakan outer query, dibuat untuk menampilkan nama, gaji dan nomor departemen pegawai yang gajinya sama dengan hasil output dari inner query.
  - Inner query akan dijalankan terlebih dahulu. Hasil dari inner query akan digunakan untuk memproses kondisi yang dijalankan pada outer query.

- Sintaks querynya adalah:

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



```
SELECT last_name, salary, department_id
FROM employees
WHERE salary IN
      (2100, 2500, 4200, 4400, 6000, 6100, 6500, 6900, 7000, 8300, 10000,
       17000);
```

LAST_NAME	SALARY	DEPARTMENT_ID
Kochhar	17000	90
De Haan	17000	90
Ernst	6000	60
Lorentz	4200	60
Popp	6900	100
Colmenares	2500	30
Vollman	6500	50
Marlow	2500	50
Olson	2100	50
Patel	2500	50
Vargas	2500	50

2. Munculkan pegawai yang bukan seorang IT Programmer (IT\_PROG) dan gajinya lebih kecil dari setiap IT programmer yang ada.
  - Untuk menyelesaikan kasus ini, dapat digunakan operator ANY.
  - Operator ANY digunakan untuk membandingkan suatu nilai untuk setiap nilai yang dikembalikan oleh subquery (inner query).
  - Penyelesaian dari soal, terdapat dua buah query. Query pertama yang merupakan inner query, dibuat untuk mencari salary dari setiap pegawai yang memiliki job 'IT\_PROG'.
  - Query kedua yang merupakan outer query, dibuat untuk mencari nomor, nama, job serta salary pegawai yang salarynya lebih kecil dari output yang dihasilkan oleh inner query.

- Sintaks querynya adalah:

```

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';
    
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
132	Olson	ST_CLERK	2100
136	Philtanker	ST_CLERK	2200
128	Markle	ST_CLERK	2200
135	Gee	ST_CLERK	2400
127	Landry	ST_CLERK	2400
191	Perkins	SH_CLERK	2500
182	Sullivan	SH_CLERK	2500
144	Vargas	ST_CLERK	2500
140	Patel	ST_CLERK	2500
131	Marlow	ST_CLERK	2500
119	Colmenares	PU_CLERK	2500

#### CATATAN !!

<ANY, berarti lebih kecil dari nilai maksimum.  
>ANY, berarti lebih dari nilai minimum  
=ANY, berarti sama dengan klausa IN  
  
>ALL, berarti lebih besar dari nilai maksimum  
<ALL, berarti lebih kecil dari nilai minimum

3. Munculkan nomor pegawai, nomor manager, nomor departemen untuk pegawai-pegawai yang memiliki manager dan bekerja dalam departemen yang sama dengan pegawai 178 dan 174. Kerjakan dengan menggunakan *Pairwise Comparison Subquery*.
  - Persoalan diatas diselesaikan dengan menggunakan **multiple-column subquery** karena subquery mengembalikan lebih dari satu kolom. Hal tersebut ditandai dengan perbandingan nilai pada kolom MANAGER\_ID dan kolom DEPARTMENT\_ID untuk setiap baris dalam table EMPLOYEES, dimana nilai yang dibandingkan adalah nilai pada kolom MANAGER\_ID dan kolom DEPARTMENT\_ID bagi pegawai 178 atau 174.

- Inner query, diciptakan untuk mendapatkan informasi MANAGER\_ID dan DEPARTMENT\_ID bagi pegawai 178 atau 174. Hasil dari inner query tersebut kemudian dibandingkan dengan nilai kolom MANAGER\_ID dan kolom DEPARTMENT\_ID dalam table EMPLOYEES. Apabila nilai yang dibandingkan sesuai maka akan dimunculkan. Record yang EMPLOYEE\_ID 178 atau 174 sendiri tidak dimunculkan.
- Berikut sintaks querynya:

```
SELECT employee_id, manager_id, department_id
FROM employees
WHERE (manager_id, department_id) IN
      (SELECT manager_id, department_id
       FROM employees
       WHERE employee_id IN (178, 174))
AND employee_id NOT IN (178, 174);
```

149, 80

EMPLOYEE_ID	MANAGER_ID	DEPARTMENT_ID
175	149	80
176	149	80
177	149	80
179	149	80

- Soal sama dengan no.3, hanya kerjakan dengan menggunakan Nonpairwise Comparison Subquery.
  - Persoalan sama dengan soal no. 3, namun diminta diselesaikan dengan menggunakan Nonpairwise Comparison Subquery.
  - Dalam nonpairwise comparison subquery, terdapat 3 query, 2 inner query dan 1 outer query.
  - Inner query yang pertama dibuat untuk mencari nilai MANAGER\_ID dari pegawai 178 dan 174.
  - Inner query yang kedua dibuat untuk mencari nilai DEPARTMENT\_ID dari pegawai 178 dan 174.
  - Kedua hasil inner query tersebut akan dikondisikan dengan outer query yang menampilkan EMPLOYEE\_ID, MANAGER\_ID serta DEPARTMENT\_ID yang memenuhi nilai-nilai yang dihasilkan dalam inner querynya.
  - Sintaks querynya adalah:

```

SELECT employee_id, manager_id, department_id
FROM employees
WHERE manager_id IN
    (SELECT manager_id
     FROM employees
     WHERE employee_id IN (178,174))
AND department_id IN
    (SELECT department_id
     FROM employees
     WHERE employee_id IN (178,174))
AND employee_id NOT IN (178,174);

```

EMPLOYEE_ID	MANAGER_ID	DEPARTMENT_ID
179	149	80
177	149	80
176	149	80
175	149	80

#### 9.4 TEST AKHIR

- Munculkan nomor departemen, last name dan id job untuk setiap pegawai yang ada pada departemen Executive. Gunakan subquery !

DEPARTMENT_ID	LAST_NAME	JOB_ID
90	King	AD_PRES
90	Kochhar	AD_VP
90	De Haan	AD_VP

- Dengan menggunakan subquery, munculkan nomor pegawai dan nama (last name) semua pegawai yang bekerja dalam departemen yang sama dengan setiap pegawai yang namanya mengandung huruf 'u'.

EMPLOYEE_ID	LAST_NAME
107	Lorentz
106	Pataballa
105	Austin
104	Ernst
103	Hunold
199	Grant
198	OConnell
197	Feeney
196	Walsh
195	Jones
194	McCain

:

3. Dengan menggunakan subquery, munculkan nama, nomor departemen dan id job untuk semua pegawai yang berada di departemen 1700.

LAST_NAME	DEPARTMENT_ID	JOB_ID
King	90	AD_PRES
Kochhar	90	AD_VP
De Haan	90	AD_VP
Greenberg	100	FI_MGR
Faviet	100	FI_ACCOUNT
Chen	100	FI_ACCOUNT
Sciarra	100	FI_ACCOUNT
Urman	100	FI_ACCOUNT
Popp	100	FI_ACCOUNT
Raphaely	30	PU_MAN
Khoo	30	PU_CLERK

:

4. Modifikasi soal no.2, untuk menampilkan nomor pegawai, nama pegawai dan salary untuk semua pegawai yang gajinya diatas rata-rata gaji dan bekerja dalam departemen yang sama dengan setiap pegawai yang namanya mengandung huruf 'u'.

EMPLOYEE_ID	LAST_NAME	SALARY
103	Hunold	9000
123	Vollman	6500
122	Kaufling	7900
121	Fripp	8200
120	Weiss	8000
177	Livingston	8400
176	Taylor	8600
175	Hutton	8800
174	Abel	11000
172	Bates	7300
171	Smith	7400

:

5. Munculkan nama pegawai (last name), nomor departemen dan salary setiap pegawai yang nomor departemen dan salarynya sama dengan nomor departemen dan salary setiap pegawai yang mendapat komisi.

LAST_NAME	DEPARTMENT_ID	SALARY
Russell	80	14000
Partners	80	13500
Errazuriz	80	12000
Abel	80	11000
Cambrault	80	11000
Vishney	80	10500
Zlotkey	80	10500
Bloom	80	10000
King	80	10000
Tucker	80	10000
Greene	80	9500

:

6. Munculkan nama pegawai (last name), nama departemen dan gaji dari setiap pegawai dimana gaji dan komisinya sama dengan gaji dan komisi dari setiap pegawai yang lokasi kerjanya memiliki id 1700.

LAST_NAME	DEPARTMENT_NAME	SALARY
King	Executive	24000
De Haan	Executive	17000
Kochhar	Executive	17000
Higgins	Accounting	12008
Greenberg	Finance	12008
Faviet	Finance	9000
Hunold	IT	9000
Chen	Finance	8200
Fripp	Shipping	8200
Sciarra	Finance	7700
Urman	Finance	7800

:

7. Buat query untuk menampilkan pegawai yang mendapatkan gaji lebih besar dari gaji semua sales manager (JOB\_ID = 'SA\_MAN'). Urutkan data berdasarkan salary dari tinggi ke rendah.

LAST_NAME	JOB_ID	SALARY
King	AD_PRES	24000
De Haan	AD_VP	17000
Kochhar	AD_VP	17000

## 10 BAB X SCALAR & CORRELATED SUBQUERY

### 10.1 IDENTITAS

#### Kajian

Mengerjakan Subquery

#### Topik

1. Scalar Subquery
2. Correlated Subquery

#### Referensi

1. Gavin Powell, C.M.D.(2005). Oracle SQL Jumpstart With Examples. USA: Elsevier Inc.
2. Greenberg, N. (Edition 1.1 August 2004). Oracle Database 10g: SQL Fundamental II. Jobi Varghese.

#### Kompetensi Utama

1. Mahasiswa mampu menggambarkan tipe persoalan yang dapat dipecahkan dengan menggunakan scalar subquery
2. Mahasiswa mampu menggambarkan tipe persoalan yang dapat dipecahkan dengan menggunakan correlated subquery
3. Mahasiswa mampu menulis scalar subquery
4. Mahasiswa mampu menulis correlated subquery

#### Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 120 menit
2. Kegiatan Mandiri : 2 x 120 menit

#### Parameter Penilaian

1. Jurnal Pengamatan
2. Tugas Akhir

## 10.2 PERTANYAAN PENDAHULUAN

1. Apa yang disebut dengan Scalar Subquery ? Jelaskan !
2. Apa yang disebut dengan Correlated Subquery ? Jelaskan !

## 10.3 PRAKTIK

### 10.3.1 Soal

1. Buat query untuk menampilkan nomor pegawai, nama pegawai (last name) dan nama departemen dari semua pegawai. Catatan: Gunakan scalar subquery untuk mengambil data nama departemen dalam klausa SELECT.
2. Munculkan seluruh pegawai yang gajinya diatas rata-rata gaji pada departemen tempat mereka bekerja.
3. Munculkan informasi mengenai pegawai yang pernah berganti job sedikitnya dua kali.

#### 10.3.1.1 Langkah Penyelesaian

1. Buat query untuk menampilkan nomor pegawai, nama pegawai (last name) dan nama departemen dari semua pegawai. Catatan: Gunakan scalar subquery untuk mengambil data nama departemen dalam klausa SELECT.
  - Scalar subquery adalah subquery yang mengembalikan hanya satu nilai kolom dari satu baris.
  - Dalam soal jelas dinyatakan bahwa scalar diletakkan dalam klausa SELECT untuk mengambil data nama departemen.
  - Sintaks querynya adalah:

```
SELECT employee_id, last_name,
       (SELECT department_name
        FROM departments d
        WHERE e.department_id = d.department_id)
  department
FROM employees e
ORDER BY department;
```

EMPLOYEE_ID	LAST_NAME	DEPARTMENT
205	Higgins	Accounting
206	Gietz	Accounting
200	Whalen	Administration
100	King	Executive
101	Kochhar	Executive
102	De Haan	Executive
109	Faviet	Finance
108	Greenberg	Finance
112	Urman	Finance
111	Sciarra	Finance
110	Chen	Finance

- :
2. Munculkan seluruh pegawai yang gajinya diatas rata-rata gaji pada departemen tempat mereka bekerja.
    - Soal diatas diselesaikan dengan menggunakan Correlated Subquery.
    - Correlated Subquery digunakan untuk pemrosesan baris per baris. Tiap-tiap subquery dijalankan sekali untuk setiap baris dari outer query.
    - Sintaks querynya adalah:

```
SELECT last_name, salary, department_id
FROM employees outer
WHERE salary > (SELECT AVG(salary)
                  FROM employees
                  WHERE department_id = outer.department_id);
```

Setiap saat baris dari outer query diproses, maka inner query dievaluasi



LAST_NAME	SALARY	DEPARTMENT_ID
King	24000	90
Hunold	9000	60
Ernst	6000	60
Greenberg	12008	100
Faviet	9000	100
Raphaely	11000	30
Weiss	8000	50
Fripp	8200	50
Kaufling	7900	50
Vollman	6500	50
Mourgos	5800	50

:

#### 10.4 TEST AKHIR

1. Dengan menggunakan correlated subquery, buat query untuk menampilkan seluruh pegawai yang gajinya lebih besar dari rata-rata gaji departemen tempat mereka bekerja. Munculkan last name, salary, department id, dan rata-rata salary departemen. Urutkan secara ascending berdasarkan rata-rata salary.

ENAME	SALARY	DEPTNO	DEPT_AVG
Fripp	8200	50	3475.55556
Chung	3800	50	3475.55556
Kaufling	7900	50	3475.55556
Mourgos	5800	50	3475.55556
Bell	4000	50	3475.55556
Rajs	3500	50	3475.55556
Everett	3900	50	3475.55556
Sarchand	4200	50	3475.55556
Bull	4100	50	3475.55556
Vollman	6500	50	3475.55556
Ladwig	3600	50	3475.55556

:

2. Dengan menggunakan scalar subquery, munculkan nama pegawai, nama departemen, gaji serta selisih maksimum gaji dengan gaji pegawai untuk pegawai-pegawai yang bekerja di departemen 50. Urutkan menaik berdasarkan gaji.

LAST_NAME	DEPARTMENT	SALARY	MAXSAL	DIFF
Olson	Shipping	2100	24000	21900
Markle	Shipping	2200	24000	21800
Philtanker	Shipping	2200	24000	21800
Gee	Shipping	2400	24000	21600
Landry	Shipping	2400	24000	21600
Marlow	Shipping	2500	24000	21500
Patel	Shipping	2500	24000	21500
Perkins	Shipping	2500	24000	21500
Sullivan	Shipping	2500	24000	21500
Vargas	Shipping	2500	24000	21500
Grant	Shipping	2600	24000	21400

:

## 11 DAFTAR PUSTAKA

- [1] Gavin Powell, C.M.D.(2005). Oracle SQL Jumpstart With Examples. USA: Elsevier Inc.
- [2] Greenberg, N. (Edition 1.1 August 2004). Oracle Database 10g: SQL Fundamental I. Jobi Varghese.
- [3] Greenberg, N. (Edition 1.1 August 2004). Oracle Database 10g: SQL Fundamental II. Jobi Varghese.