Tugas Pertemuan 10.1 – Introduction PL/SQL

Mata Kuliah: Basis Data (2023)

Prodi: D3-Teknik Informatika POLBAN

Kelas: 2A

NIM: 221511026

Nama Lengkap: Paulina Lestari Simatupang

Prasyarat

1. Menggunakan alat bantu atau perangkat lunak RDBMS Oracle
2. Menggunakan skema HR Oracle.

Instruksi

1. Total seluruh soal yang bisa dikerjakan sebanyak **3 soal**.
2. **Disarankan mengerjakan soal secara terturut**!
3. Sertakan jawaban dalam bentuk syntax PL/SQL!
4. Sertakan **screenshot halaman penuh yang menyertakan taskbar dengan waktu** (Print Screen) untuk menunjukkan syntax dan hasil eksekusi query pada RDBMS Oracle!
5. Ekspor file lembar jawaban ini dalam format PDF dengan nama file:
   BD_P_10_1_<Kelas>_<NIM>_<NamaLengkap>.pdf
   contoh sebagai berikut:
   BD_P_10_1_2A_221511036_AdrianEkaSaputra.pdf
6. Kumpulkan melalui tautan Google Form yang dikirimkan melalui WhatsApp Group sesuai dengan tenggat waktu yang telah ditentukan!

Referensi

1. https://bit.ly/oracle-plsql
2. https://www.oracletutorial.com/plsql-tutorial/
3. https://docs.oracle.com/en/database/oracle/oracle-database/19/lnpls/database-pl-sql-language-reference.pdf

Jika ada hal yang ingin ditanyakan, silahkan sampaikan melalui WhatsApp Group.

Selamat mengerjakan!

Soal No. 1

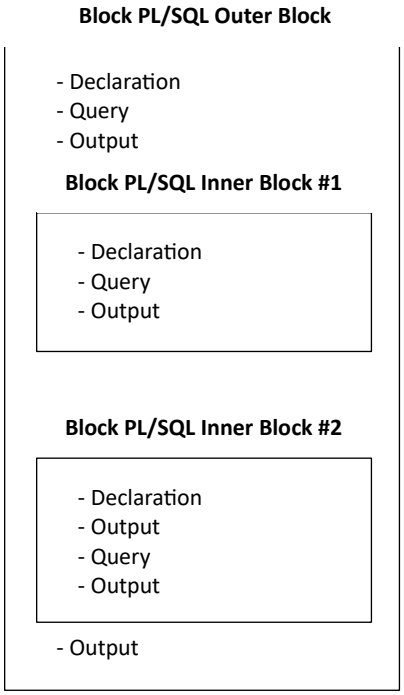
Soal

a. Tampilkan hanya nilai `take_home_pay` dari tabel `employees` sesuai dengan ketentuan berikut:
   - Mendeklarasikan variabel `take_home_pay` dengan tipe data yang sesuai menurut Anda!
   - Melakukan query hanya menampilkan satu kolom `take_home_pay` dengan `employee_id` = 120!
   - Kolom baru `take_home_pay` pada query diperoleh dari `salary` dikali `commission_pct`! Jika nilai `commission_pct` adalah NULL, maka nilai `take_home_pay` sama seperti `salary`! Hasil akhir `take_home_pay` tidak boleh NULL menggunakan function NVL()!
   - Hasil query tersebut disimpan terlebih dahulu pada variabel `take_home_pay`!
   - Menampilkan nilai dari variabel `take_home_pay` dengan menggunakan function DBMS_OUTPUT.PUT_LINE() diawali dengan teks "Take Home Pay for Employee with ID 120 = "!

Contoh Output 1A:

Take Home Pay for Employee with ID 120 = 8000

b. Tampilkan gaji employee terendah dan tertinggi dari tabel `employees` sesuai dengan ketentuan berikut:



Block PL/SQL Outer Block

- Mendeklarasikan empat variabel pada block PL/SQL **Outer Block** dengan tipe data yang sesuai menurut Anda!
  - `lowest_salary`
  - `emp_name_lowest_salary`
  - `highest_salary`
  - `emp_name_highest_salary`
- Melakukan query untuk memperoleh data hanya satu employee dengan gaji terendah, dengan menampilkan kolom `full_name` (gabungan dari `first_name` dan `last_name`)

dan kolom `lowest_salary` (menggunakan fungsi agregasi dari kolom `salary`)! Query harus dilakukan hanya satu kali, tidak boleh lebih!

- Hasil query tersebut disimpan terlebih dahulu pada variabel `lowest_salary` untuk kolom `lowest_salary` dan variabel `emp_name_lowest_salary` untuk kolom `full_name`!
- Menampilkan nilai dari keempat variabel tersebut dengan menggunakan function DBMS_OUTPUT.PUT_LINE() sesuai dengan **Contoh Output** berikut:

```
LOWEST SALARY
Employee Full Name: TJ Olson
Salary: 2.100

HIGHEST SALARY
Employee Full Name: Steven King
Salary: 24.000
```

Block PL/SQL Inner Block #1

- Mendeklarasikan empat variabel baru dengan nama yang sama persis pada block PL/SQL **Inner Block #1**!
- Melakukan query untuk memperoleh data hanya satu employee dengan gaji terendah **yang bukan merupakan President dan Vice President**, dengan menampilkan kolom `full_name` (gabungan dari `first_name` dan `last_name`) dan kolom `lowest_salary` (menggunakan fungsi agregasi dari kolom `salary`)! Query harus dilakukan hanya satu kali, tidak boleh lebih!
- Hasil query tersebut disimpan terlebih dahulu pada variabel `lowest_salary` untuk kolom `lowest_salary` dan variabel `emp_name_lowest_salary` untuk kolom `full_name`!
- Menampilkan nilai dari keempat variabel tersebut dengan menggunakan function DBMS_OUTPUT.PUT_LINE() sesuai dengan **Contoh Output** berikut:

```
LOWEST SALARY
Employee Full Name: TJ Olson
Salary: 2.100

HIGHEST SALARY
Employee Full Name: John Rusell
Salary: 14.000
```

Block PL/SQL Inner Block #2

- Mendeklarasikan empat variabel baru dengan nama yang sama persis pada block PL/SQL **Inner Block #2**!
- Menampilkan nilai dari keempat variabel tersebut dengan menggunakan function DBMS_OUTPUT.PUT_LINE()!
- Melakukan query untuk memperoleh data hanya satu employee dengan gaji terendah **yang bukan merupakan President, Vice President dan Seluruh Clerk (terdapat 3 jenis Clerk)**, dengan menampilkan kolom `full_name` (gabungan dari `first_name` dan `last_name`) dan kolom `lowest_salary` (menggunakan fungsi agregasi dari kolom `salary`)! Query harus dilakukan hanya satu kali, tidak boleh lebih!
- Hasil query tersebut disimpan terlebih dahulu pada variabel `lowest_salary` untuk kolom `lowest_salary` dan variabel `emp_name_lowest_salary` untuk kolom `full_name`!
- Menampilkan nilai dari keempat variabel tersebut dengan menggunakan function DBMS_OUTPUT.PUT_LINE() sesuai dengan **Contoh Output** berikut:

```
LOWEST SALARY
Employee Full Name: Diana Lorentz
Salary: 4.200

HIGHEST SALARY
```

Employee Full Name: John Rusell
Salary: 14.000

Block PL/SQL Outer Block

- Pada posisi paling bawah setelah selesai block PL/SQL Inner Block #2, tampilkan ulang nilai dari keempat variabel tersebut dengan menggunakan function DBMS_OUTPUT.PUT_LINE() sesuai dengan **Contoh Output** berikut:

LOWEST SALARY
Employee Full Name: TJ Olson
Salary: 2.100

HIGHEST SALARY
Employee Full Name: Steven King
Salary: 24.000

Jawaban (Penjelasan dan Syntax)

a.
```
SELECT NVL(salary * NVL(commission_pct, 1), salary) AS take_home_pay
FROM employees
WHERE employee_id = 120;
```

Pada kode di atas menggunakan NVL untuk menangani nilai null dalam 'salary' dan 'commission_pct'. Fungsi NVL menerima dua argument yaitu nilai yang akan diuji apakah null atau tidak, dan nilai kedua adalah nilai yang akan digunakan jika nilai pertama null. Jika commission_pct tidak NULL, maka akan mengalikan salary dengan commission_pct, jika commission_pct NULL, maka akan menggantikan commission_pct dengan 1 dan mengalikan salary dengan hasilnya atau hanya menggunakan salary jika commission_pct NULL.

	TAKE_HOME_PAY
1	8000

(kolom `take_home_pay` dengan `employee_id` = 120)

```
DECLARE
  take_home_pay NUMBER;
BEGIN
  SELECT NVL(salary, 0) * NVL(commission_pct, 1)
  INTO take_home_pay
  FROM employees
  WHERE employee_id = 120;
  DBMS_OUTPUT.PUT_LINE('Take Home Pay for Employee with ID 120 = ' ||
take_home_pay);
END;
/
```

‘Declare’ merupakan bagian awal dari pml/sql dimana mendeklarasikan variable dan tipe data yang akan digunakan dalam blok pl/sql. Pada kode di atas mendeklarasikan variable ‘take_home_pay’ dengan tipe data number. ‘Begin’ merupa kan blok awal pl/sql yang akan menjalankan pernyataan-pernyataan di dalamnya. Jika kolom ‘salary’ null maka NVL(salary, 0) digunakan untuk menggantikan nilai NULL dalam kolom salary dengan 0. Jika kolom ‘commission_pct’ null maka NVL(commission_pct, 1) digunakan untuk menggantikan nilai NULL dalam kolom commission_pct dengan 1. Hasil perhitungan tersebut digunakan untuk karyawan dengan ‘employee_id’ = 120.

DBMS_OUTPUT.PUT_LINE('Take Home Pay for Employee with ID 120 = ' || take_home_pay); digunakan untuk menampilkan hasil ke layar. Dan ‘End’ menandakan akhir dari blok pl/sql.

b. **Block PL/SQL Outer Block**

```
DECLARE
   lowest_salary NUMBER;
   emp_name_lowest_salary VARCHAR2(100);
   highest_salary NUMBER;
   emp_name_highest_salary VARCHAR2(100);
BEGIN
   SELECT MIN(salary), MAX(salary)
   INTO lowest_salary, highest_salary
   FROM employees;

   SELECT first_name || ' ' || last_name
   INTO emp_name_lowest_salary
   FROM employees
   WHERE salary = lowest_salary;

   SELECT first_name || ' ' || last_name
   INTO emp_name_highest_salary
   FROM employees
   WHERE salary = highest_salary;

   DBMS_OUTPUT.PUT_LINE('LOWEST SALARY');
   DBMS_OUTPUT.PUT_LINE('Employee Full Name: ' || emp_name_lowest_salary);
   DBMS_OUTPUT.PUT_LINE('Salary: ' || lowest_salary);

   DBMS_OUTPUT.PUT_LINE('HIGHEST SALARY');
   DBMS_OUTPUT.PUT_LINE('Employee Full Name: ' || emp_name_highest_salary);
   DBMS_OUTPUT.PUT_LINE('Salary: ' || highest_salary);
END;
/
```

lowest_salary NUMBER; emp_name_lowest_salary VARCHAR2(100); highest_salary NUMBER; emp_name_highest_salary VARCHAR2(100): merupakan deklarasi variabel yang digunakan untuk menyimpan nilai gaji terendah, nama karyawan dengan gaji terendah, gaji

tertinggi, dan nama karyawan dengan gaji tertinggi. SELECT MIN(salary), MAX(salary) INTO lowest_salary, highest_salary FROM employees; merupakan pernyataan yang mengambil nilai gaji terendah (MIN) dan tertinggi (MAX) dari kolom salary dalam tabel employees dan menyimpannya dalam variabel lowest_salary dan highest_salary. SELECT first_name || ' ' || last_name INTO emp_name_lowest_salary FROM employees WHERE salary = lowest_salary; merupakan pernyataan yang mengambil nama lengkap karyawan dengan gaji terendah dan menyimpannya dalam variabel emp_name_lowest_salary. Pernyataan ini mencari karyawan yang memiliki gaji yang sama dengan lowest_salary. SELECT first_name || ' ' || last_name INTO emp_name_highest_salary FROM employees WHERE salary = highest_salary; merupakan pernyataan ketiga yang melakukan hal yang sama dengan pernyataan kedua, tetapi untuk karyawan dengan gaji tertinggi.

Block PL/SQL Inner Block #2

```
DECLARE
   lowest_salary NUMBER;
   emp_name_lowest_salary VARCHAR2(100);
   highest_salary NUMBER;
   emp_name_highest_salary VARCHAR2(100);

BEGIN
   SELECT MIN(salary), MAX(salary)
   INTO lowest_salary, highest_salary
   FROM employees
   WHERE job_id NOT IN ('PRESIDENT', 'VICE PRESIDENT', 'CLERK');

   SELECT first_name || ' ' || last_name
   INTO emp_name_lowest_salary
   FROM employees
   WHERE salary = lowest_salary
   AND job_id NOT IN ('PRESIDENT', 'VICE PRESIDENT', 'CLERK');

   SELECT first_name || ' ' || last_name
   INTO emp_name_highest_salary
   FROM employees
   WHERE salary = highest_salary
   AND job_id NOT IN ('PRESIDENT', 'VICE PRESIDENT', 'CLERK');

   DBMS_OUTPUT.PUT_LINE('LOWEST SALARY');
   DBMS_OUTPUT.PUT_LINE('Employee Full Name: ' || emp_name_lowest_salary);
   DBMS_OUTPUT.PUT_LINE('Salary: ' || lowest_salary);

   DBMS_OUTPUT.PUT_LINE('HIGHEST SALARY');
   DBMS_OUTPUT.PUT_LINE('Employee Full Name: ' || emp_name_highest_salary);
   DBMS_OUTPUT.PUT_LINE('Salary: ' || highest_salary);
END;
```

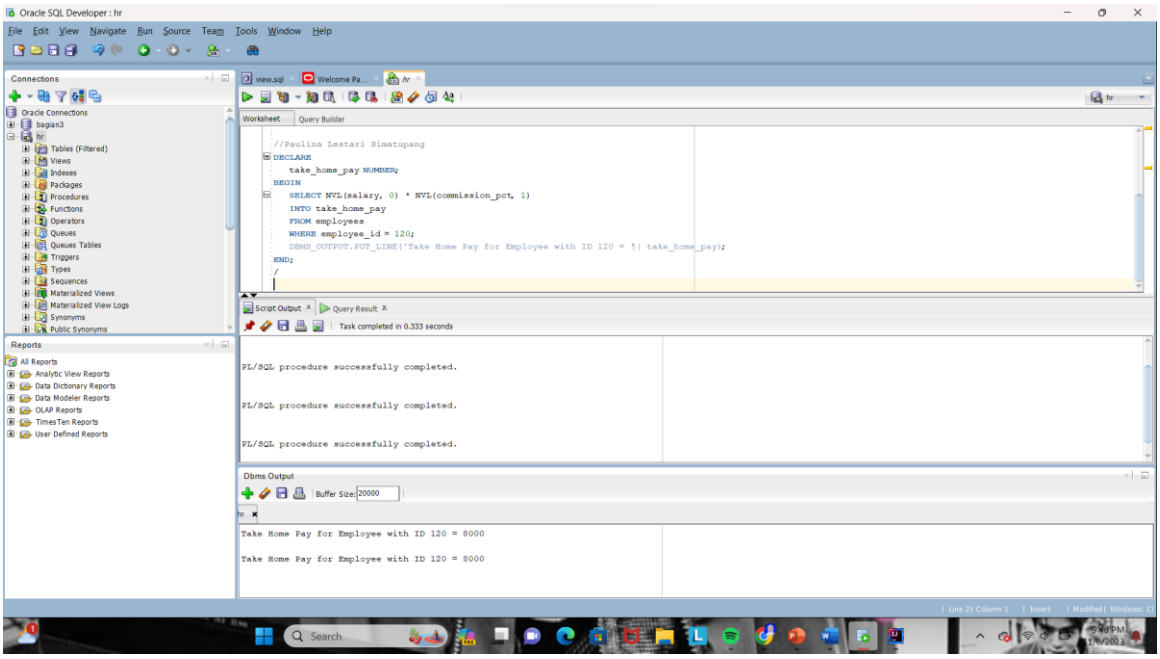
Block PL/SQL Outer Block

```
DECLARE
   lowest_salary NUMBER;
   emp_name_lowest_salary VARCHAR2(100);
   highest_salary NUMBER;
   emp_name_highest_salary VARCHAR2(100);

BEGIN
   -- Query untuk mendapatkan gaji terendah yang bukan President, Vice President, atau
Clerk
   SELECT MIN(salary)
   INTO lowest_salary
   FROM employees
   WHERE job_id NOT IN ('PRESIDENT', 'VICE PRESIDENT', 'CLERK');

   -- Query untuk mendapatkan nama karyawan dengan gaji terendah
   SELECT first_name || ' ' || last_name
   INTO emp_name_lowest_salary
   FROM employees
   WHERE salary = lowest_salary
   AND job_id NOT IN ('PRESIDENT', 'VICE PRESIDENT', 'CLERK');

   -- Menampilkan hasil Inner Block #1
   DBMS_OUTPUT.PUT_LINE('LOWEST SALARY');
   DBMS_OUTPUT.PUT_LINE('Employee Full Name: ' || emp_name_lowest_salary);
   DBMS_OUTPUT.PUT_LINE('Salary: ' || lowest_salary);

   -- Menampilkan ulang hasil Inner Block #1
   DBMS_OUTPUT.PUT_LINE('LOWEST SALARY');
   DBMS_OUTPUT.PUT_LINE('Employee Full Name: ' || emp_name_lowest_salary);
   DBMS_OUTPUT.PUT_LINE('Salary: ' || lowest_salary);

   -- Menampilkan ulang hasil Inner Block #1
   DBMS_OUTPUT.PUT_LINE('LOWEST SALARY');
   DBMS_OUTPUT.PUT_LINE('Employee Full Name: ' || emp_name_lowest_salary);
   DBMS_OUTPUT.PUT_LINE('Salary: ' || lowest_salary);

   -- Query untuk mendapatkan gaji tertinggi
   SELECT MAX(salary)
   INTO highest_salary
   FROM employees;

   -- Query untuk mendapatkan nama karyawan dengan gaji tertinggi
   SELECT first_name || ' ' || last_name
   INTO emp_name_highest_salary
   FROM employees
   WHERE salary = highest_salary;

   -- Menampilkan hasil gaji tertinggi
   DBMS_OUTPUT.PUT_LINE('HIGHEST SALARY');
   DBMS_OUTPUT.PUT_LINE('Employee Full Name: ' || emp_name_highest_salary);
```

```
    DBMS_OUTPUT.PUT_LINE('Salary: ' || highest_salary);
END;
/
```

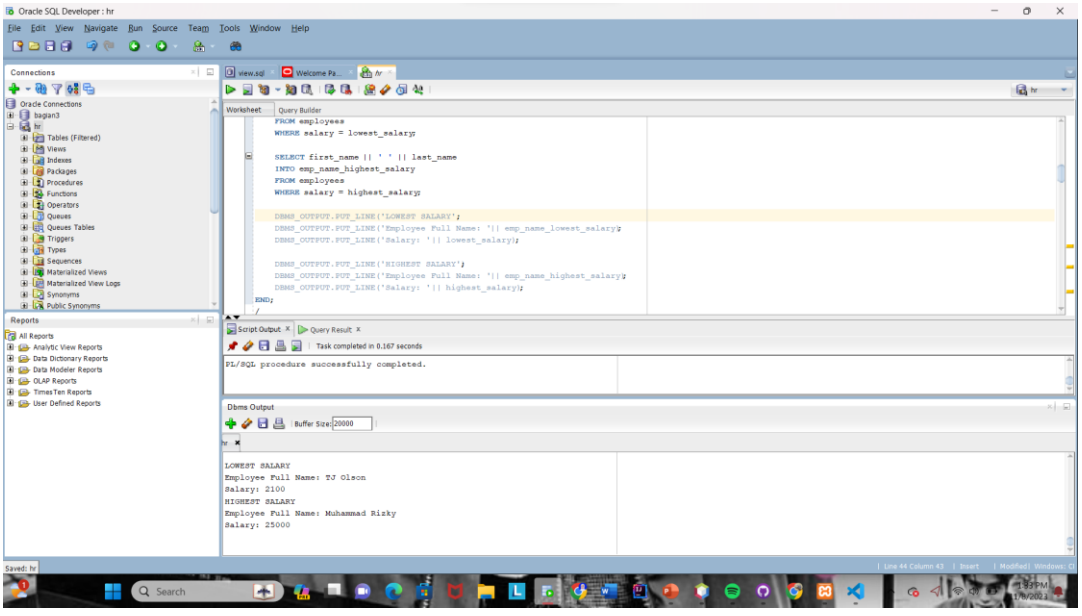
Hasil Eksekusi (Screenshot)

a.

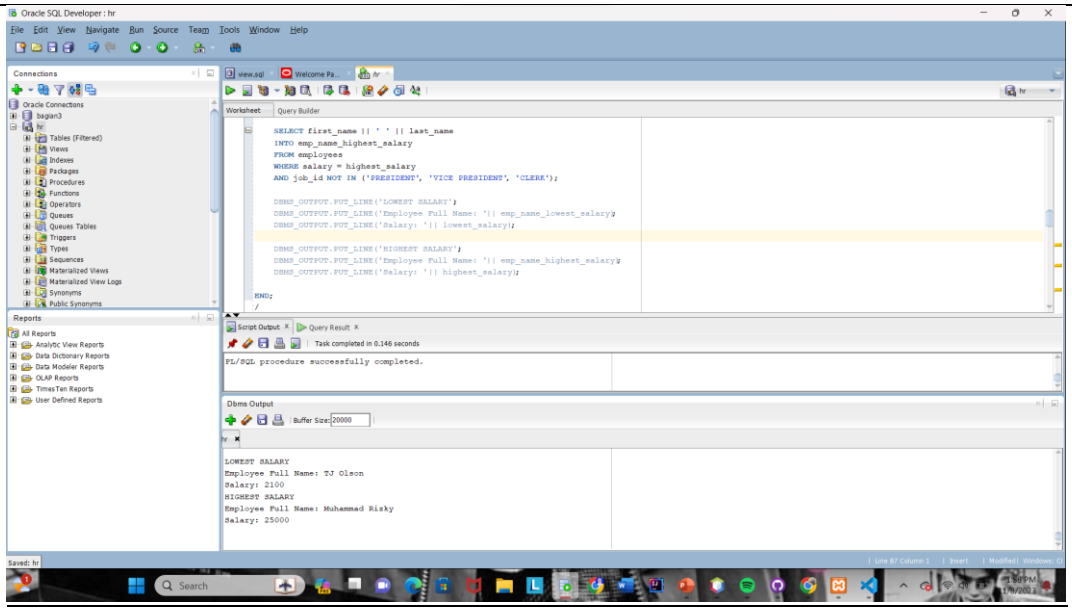


b.

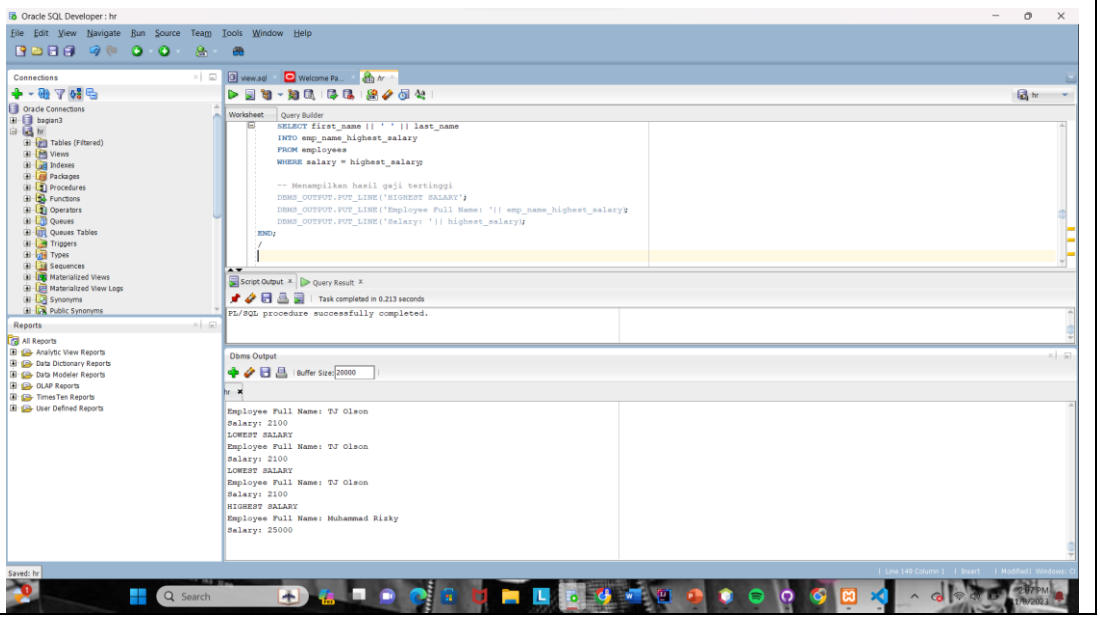
- **Block PL/SQL Outer Block**



- **Block PL/SQL Inner Block #2**



- **Block PL/SQL Outer Block**



Soal No. 2

Soal

- Lakukan query untuk memperoleh seluruh data employee yang ditampung terlebih dahulu ke dalam suatu variabel!
- Selanjutnya, tampilkan data employee tersebut menggunakan looping dan function DBMS_OUTPUT.PUT_LINE!

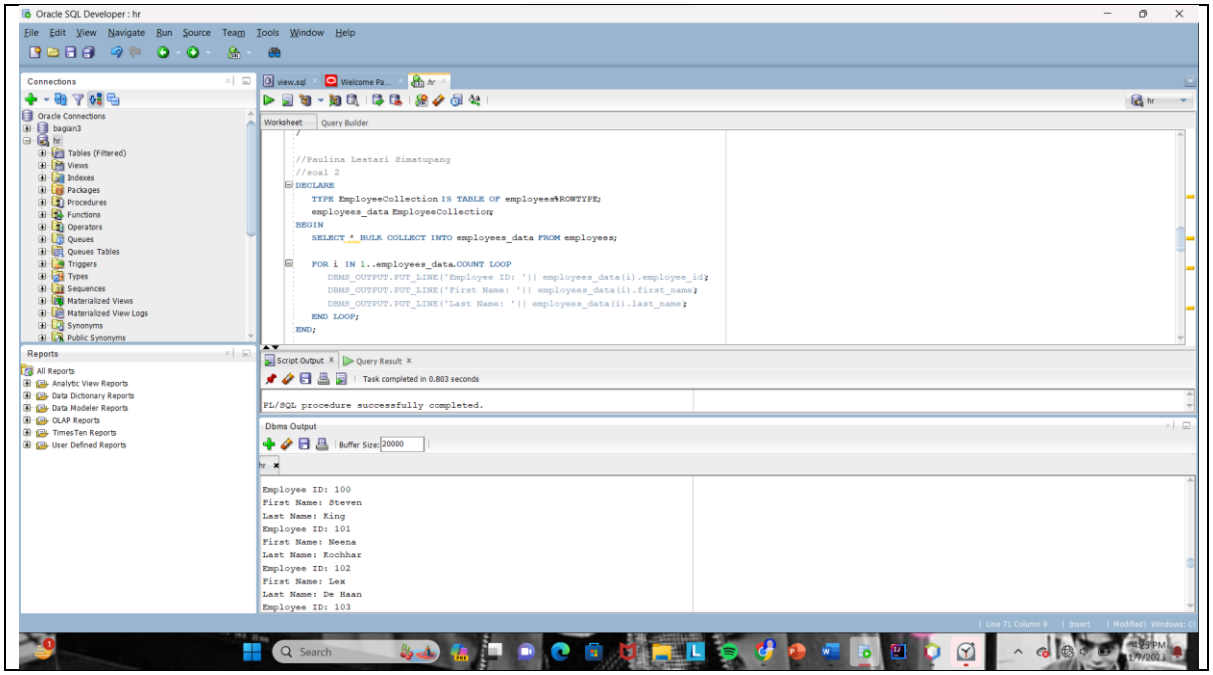
Jawaban (Penjelasan dan Syntax)

Untuk memperoleh seluruh data employee yang di tamping terlebih dahulu ke dalam satu variable dapat dilakukan menggunakan kode di bawah ini:

```
DECLARE
  TYPE EmployeeCollection IS TABLE OF employees%ROWTYPE;
  employees_data EmployeeCollection;
BEGIN
  SELECT * BULK COLLECT INTO employees_data FROM employees;
  FOR i IN 1..employees_data.COUNT LOOP
    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || employees_data(i).employee_id);
    DBMS_OUTPUT.PUT_LINE('First Name: ' || employees_data(i).first_name);
    DBMS_OUTPUT.PUT_LINE('Last Name: ' || employees_data(i).last_name);
  END LOOP;
END;
```

TYPE EmployeeCollection IS TABLE OF employees%ROWTYPE; merupakan deklarasi tipe koleksi EmployeeCollection yang berisi row dari tabel employees. employees%ROWTYPE merupakan tipe yang sesuai dengan struktur kolom dalam tabel employees. SELECT * BULK COLLECT INTO employees_data FROM employees; merupakan pernyataan yang mengambil semua data dari tabel employees dan menyimpannya dalam variabel koleksi employees_data menggunakan klausa BULK COLLECT. Dengan demikian, semua baris data dari tabel employees disimpan dalam koleksi. Klausa ‘Bulk Collect’ merupakan fitur dalam pl/sql yang digunakan untuk mengumpulkan sejumlah besar row hasil operasi select dalam satu pernyataan koleksi.

Hasil Eksekusi (Screenshot)



Soal No. 3

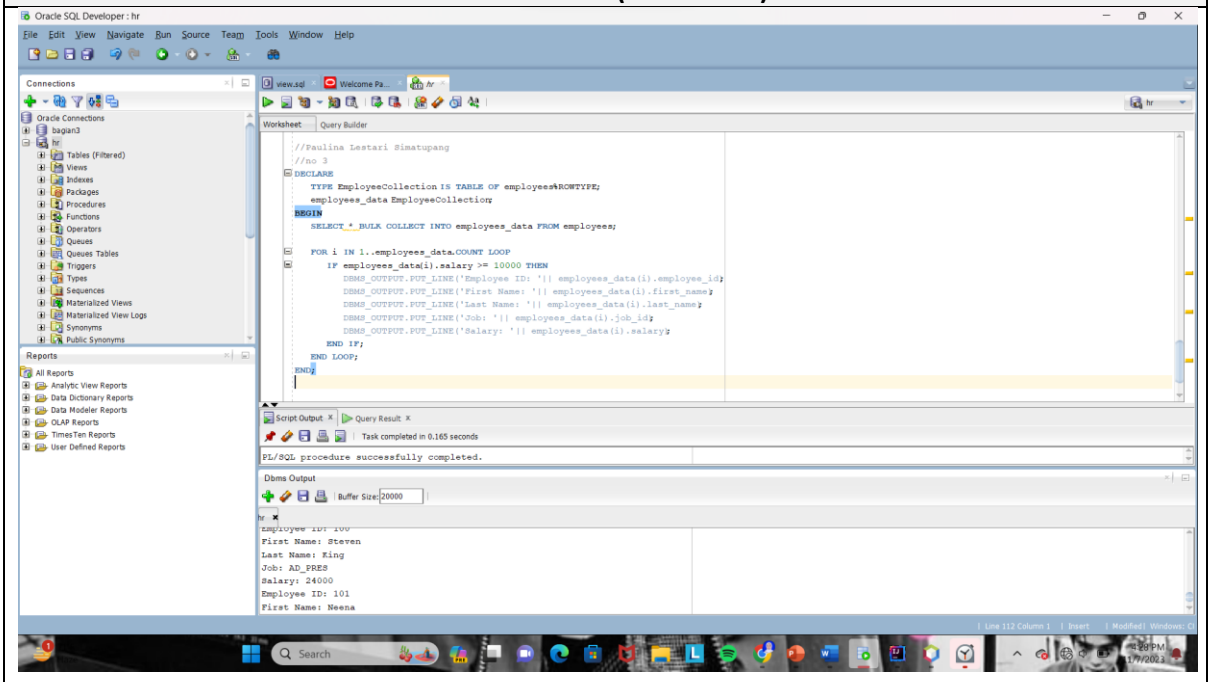
Soal
- Lakukan query untuk memperoleh seluruh data employee yang ditampung terlebih dahulu ke dalam suatu variabel! - Selanjutnya, tampilkan data employee tersebut menggunakan looping dan function DBMS_OUTPUT.PUT_LINE! - Saat melakukan looping tambahkan kondisi (IF-ELSE) hanya menampilkan data employee yang memiliki salary minimal 10.000 dan pekerjaannya bukan merupakan ST_CLERK!
Jawaban (Penjelasan dan Syntax)

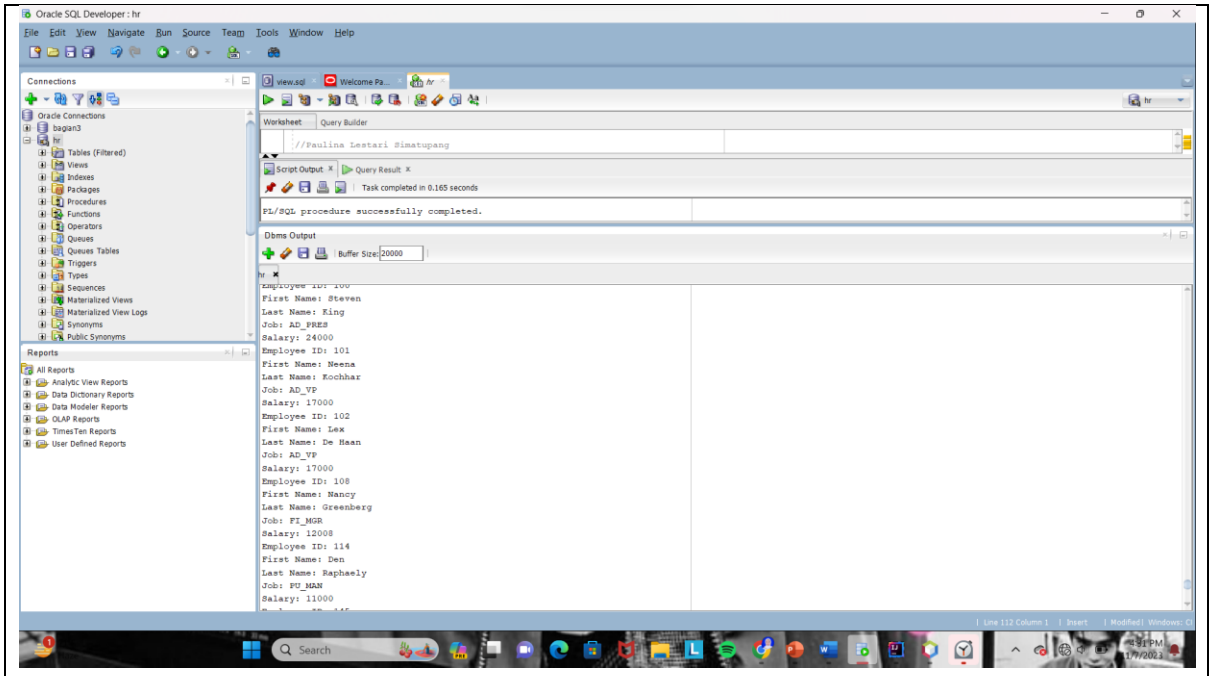
Untuk memperoleh seluruh data employee yang di tamping terlebih dahulu ke dalam suatu variable serta menggunakan looping dan function DBMS_OUTPUT.PUT_LINE! dan pada looping tersebut menambahkan konsidi (IF-ELSE) hanya menampilkan data employee yang memiliki salary minimal 10.000 dan pekerjaannya bukan merupakan ST_CLERK dapat menggunakan kode di bawah ini:
DECLARE
TYPE EmployeeCollection IS TABLE OF employees%ROWTYPE;
employees_data EmployeeCollection;
BEGIN
SELECT * BULK COLLECT INTO employees_data FROM employees;
 	
FOR i IN 1..employees_data.COUNT LOOP
IF employees_data(i).salary >= 10000 THEN
DBMS_OUTPUT.PUT_LINE('Employee ID: ' || employees_data(i).employee_id);
DBMS_OUTPUT.PUT_LINE('First Name: ' || employees_data(i).first_name);

```
        DBMS_OUTPUT.PUT_LINE('Last Name: ' || employees_data(i).last_name);
        DBMS_OUTPUT.PUT_LINE('Job: ' || employees_data(i).job_id);
        DBMS_OUTPUT.PUT_LINE('Salary: ' || employees_data(i).salary);
    END IF;
  END LOOP;
END;
```

employees_data EmployeeCollection; merupakan deklarasi variabel employees_data dengan tipe koleksi EmployeeCollection. SELECT * BULK COLLECT INTO employees_data FROM employees; merupakan pernyataan yang mengambil semua data dari tabel employees dan menyimpannya dalam variabel koleksi employees_data menggunakan klausa BULK COLLECT. Blok ini menggunakan perulangan FOR untuk mengakses setiap elemen koleksi. Perulangan ini berjalan dari 1 hingga jumlah elemen dalam koleksi (employees_data.COUNT). Pernyataan IF untuk memeriksa apakah gaji karyawan (yang diakses melalui employees_data(i).salary) setara atau lebih dari 10.000.

Hasil Eksekusi (Screenshot)




Oracle SQL Developer : hr
File Edit View Navigate Run Source Team Tools Window Help
Connections
Oracle Connections
bagian3
hr
Tables (Filtered)
Views
Indexes
Packages
Procedures
Functions
Operators
Queues
Queues Tables
Triggers
Types
Sequences
Materialized Views
Materialized View Logs
Synonyms
Public Synonyms
Reports
All Reports
Analytic View Reports
Data Dictionary Reports
Data Modeler Reports
OLAP Reports
TimesTen Reports
User Defined Reports
view.sql
Welcome Pa...
hr
Worksheet
Query Builder
//Paulina Lestari Simatupang
Script Output
Query Result
Task completed in 0.165 seconds
PL/SQL procedure successfully completed.
Dbms Output
Buffer Size: 20000
First Name: Steven
Last Name: King
Job: AD_PRES
Salary: 24000
Employee ID: 101
First Name: Neena
Last Name: Kochhar
Job: AD_VP
Salary: 17000
Employee ID: 102
First Name: Lex
Last Name: De Haan
Job: AD_VP
Salary: 17000
Employee ID: 108
First Name: Nancy
Last Name: Greenberg
Job: FI_MGR
Salary: 12008
Employee ID: 114
First Name: Den
Last Name: Raphaely
Job: PU_MAN
Salary: 11000
Search