TRƯỜNG ĐH CÔNG NGHỆ THÔNG TIN TP.HCM
Khoa Hệ Thống Thông Tin
Chương trình đào tạo: Kỹ sư Công nghệ thông tin

Bài Thực hành số 3

Tên môn học: HQTCSDL Oracle

NGÔN NGỮ PL / SQL

Bài thực hành này giúp sinh viên làm quen với các câu lệnh / khối lệnh PL / SQL.

I. Tóm tắt bài thực hành

1.1. Yêu cầu lý thuyết

Sinh viên đã được trang bị kiến thức:

- Sinh viên đã biết thao tác sử dụng Oracle và công cụ SQL Developer.
- Các kiến thức cơ bản về ngôn ngữ PL / SQL.
- Các khái niệm: sequences, views, indexs, synnonyms,.. trong Oracle.

1.2. Nội dung

❖ **Sử dụng công cụ Oracle SQL Developer 4 R2 kết nối vào CSDL Oracle 11g**

- Sinh viên xem lại phần thực hành buổi 2.
- Kết nối vào CSDL Oracle bằng user đã tạo buổi 2.

❖ **Sử dụng công cụ Oracle SQL Developer 4 R2 xây dựng các câu lệnh, khối lệnh cơ bản:**

- Các câu lệnh nhập / xuất cơ bản trong môi trường Oracle SQL Developer 4 R2.
- Khai báo biến, kiểu dữ liệu
- Các cấu trúc điều khiển
- Thủ tục / Hàm / Trigger
- Con trỏ Cursor

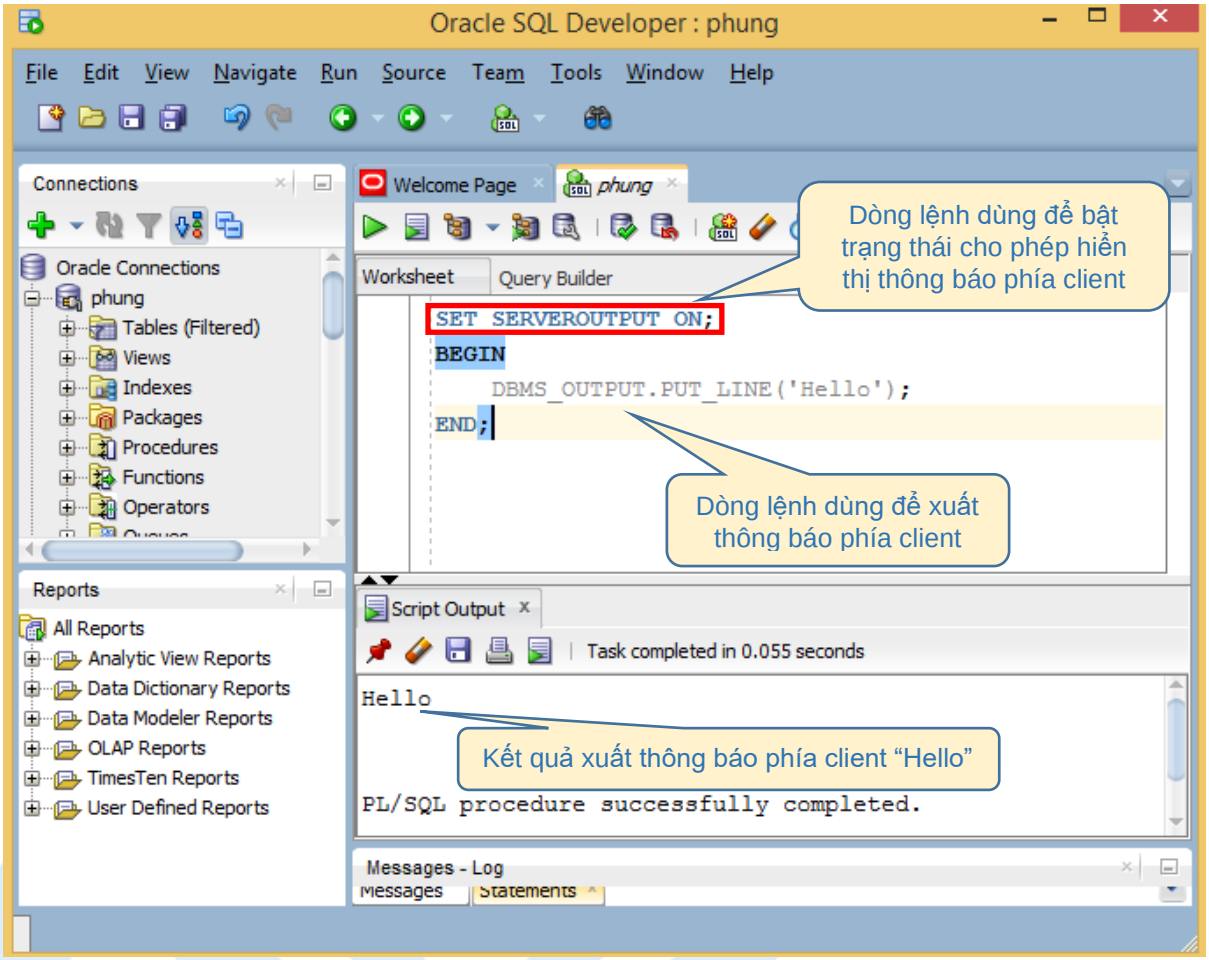
❖ **Một số lưu ý**

- Sinh viên cần nắm rõ các khái niệm liên quan đến quy tắc lập trình.

II. Thao tác từng bước

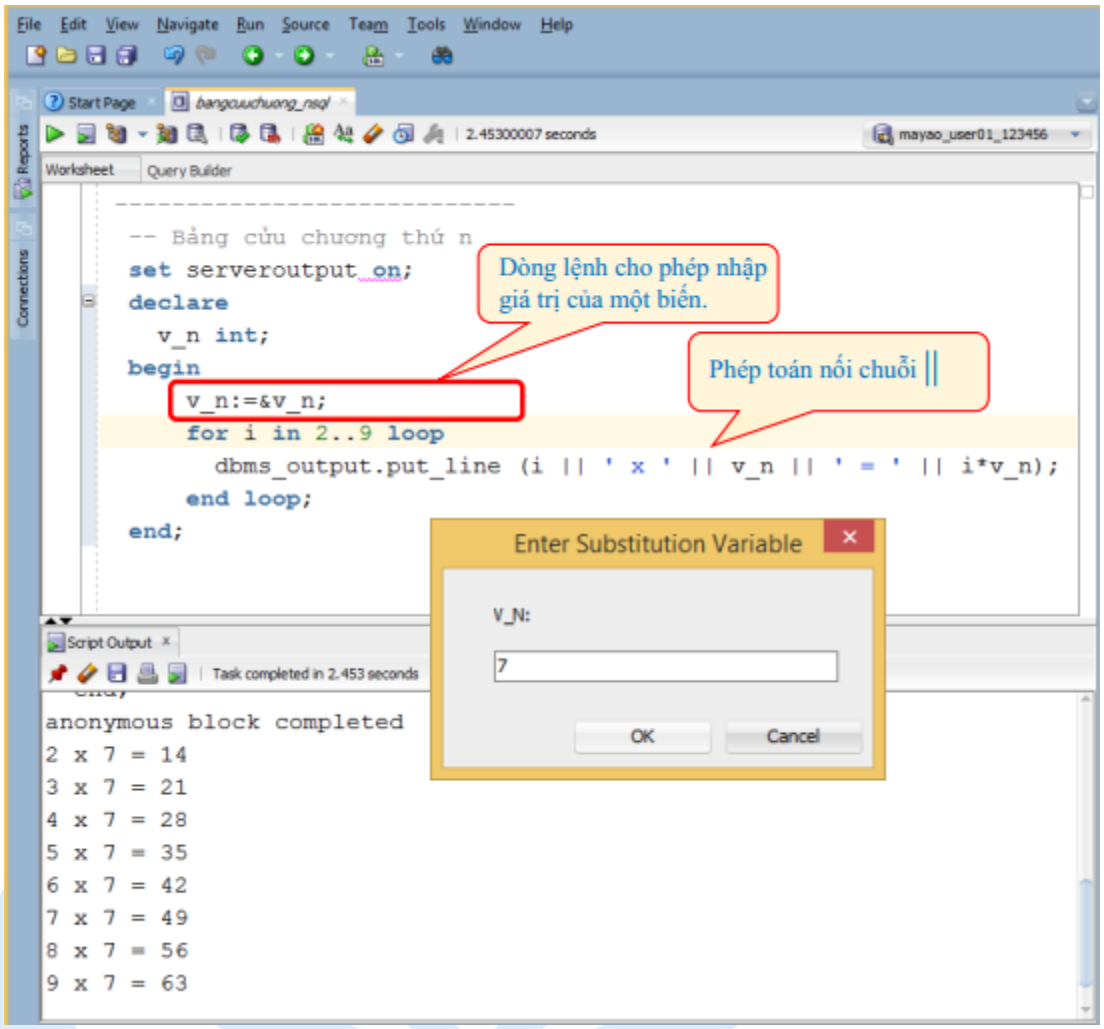
2.1. Xây dựng khối lệnh xuất ra dòng chữ "Hello world"

- Kết nối công cụ Oracle SQL Developer 4 R2 với CSDL Oracle và nhập đoạn lệnh sau:

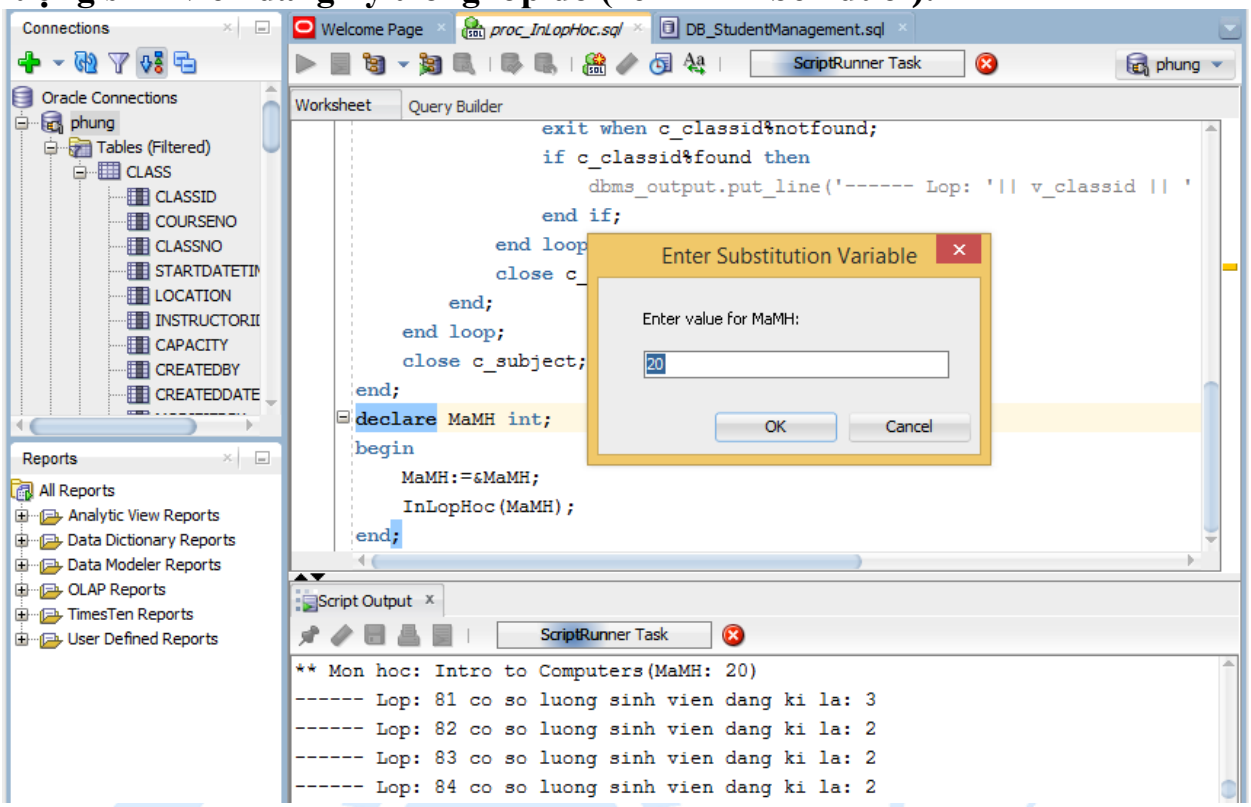


2.2. Xây dựng khối lệnh xuất bảng cửu chương thứ n (n nhập từ bàn phím)

- Kết nối công cụ Oracle SQL Developer 4 R2 với CSDL Oracle và nhập đoạn lệnh sau:



2.3. Sử dụng CSDL Quản lý sinh viên, xây dựng thủ tục cho phép nhập vào mã số môn học và in ra danh sách các lớp học của môn học này và số lượng sinh viên đăng ký trong lớp đó (xem hình bên dưới):



Nhận xét:

- Yêu cầu bài toán trên có liên quan đến việc xây dựng thủ tục **Procedure.**
- Để in ra kết quả theo yêu cầu của bài toán thì cần phải sử dụng con trỏ **Cursor**.

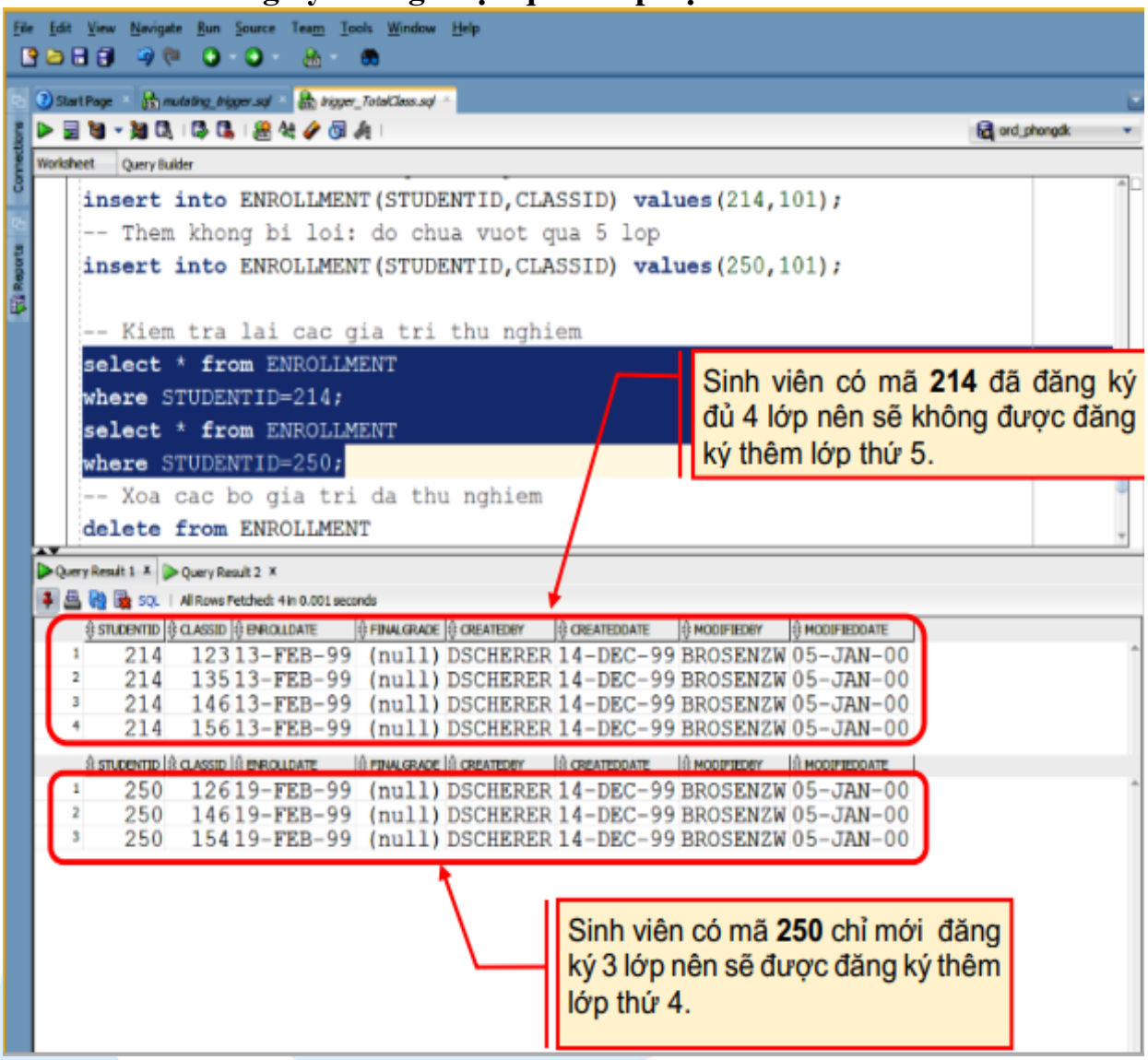
- **Viết code**

```
set serveroutput on;
create or replace procedure InLopHoc(MaMH int) is
  cursor c_subject is select courseno,description from course where courseno=MaMH;
  v_courseno course.courseno%type;
  v_description course.description%type;
begin
  open c_subject;
  loop
    fetch c_subject into v_courseno,v_description;
    exit when c_subject%notfound;
    dbms_output.put_line('** Mon hoc: ' || v_description || '(MaMH: ' ||v_courseno||')');
    declare
      cursor c_classid is select c.classid,count(*)
            from class c,enrollment e
            where c.classid=e.classid
            and c.courseno=v_courseno
            group by c.classid;
      v_classid class.classid%type;
      v_total number(10);
    begin
      open c_classid;
      loop
        fetch c_classid into v_classid,v_total;
        exit when c_classid%notfound;
        if c_classid%found then
dbms_output.put_line('-- Lop ' || v_classid || ': co so luong sinh vien dang ky la: '|| v_total);
        end if;
      end loop;
      close c_classid;
    end;
  end loop;
  close c_subject;
end;
```

- **Thực thi thủ tục**

```
declare
  MaMH int;
begin
  MaMH:=&MaMH;
  InLopHoc(MaMH);
end;
```

2.4. Sử dụng CSDL Quản lý giáo vụ, xây dựng ràng buộc trigger sao cho mỗi sinh viên đăng ký không được quá 4 lớp học:



- **Viết code**

```
set serveroutput on;
create or replace trigger trg_EnrollMent_Insert
before insert on ENROLLMENT
for each row
declare
  totalClass int;
begin
  select count(CLASSID) into totalClass
  from ENROLLMENT
  where STUDENTID=:new.STUDENTID;
  if (totalClass>=4) then
    RAISE_APPLICATION_ERROR(-20000, 'Tong so lop cua sinh vien da qua 5');
  else
    DBMS_OUTPUT.PUT_LINE('Da them thanh cong');
  end if;
end;
```

Đếm số lớp mà sinh viên này đã đăng ký

Lệnh **RAISE_APPLICATION_ERROR()** dùng để huỷ thao tác thêm khi điều kiện không thoả.

- **Thực thi kiểm tra trigger**

```
-- Them bi loi: do vuot qua 5 lop
insert into ENROLLMENT(STUDENTID,CLASSID) values(214,101);
-- Them khong bi loi: do chua vuot qua 5 lop
insert into ENROLLMENT(STUDENTID,CLASSID) values(250,101);
```

- **Nhận xét**

  - Đoạn lệnh trên chỉ mới ràng buộc cho câu lệnh Insert.
  - Cần phải viết ràng buộc trên câu lệnh Update.

2.5. Tìm hiểu và cài đặt mutating trigger và compound trigger:

- Xây dựng quan hệ Employee như sau:

```
create table employee
(
  id int primary key,
  name nvarchar2(50),
  salary int,
  id_manager int references employee(id)
);
delete from employee;
insert into employee values(1,'Phong',500,null);
insert into employee values(2,'Tri',800,1);
insert into employee values(3,'Nguyen',800,1);
insert into employee values(4,'Thanh',900,2);
select * from employee;
```

- Lần lượt cài đặt và nhận xét kết quả thực thi của các trigger cho từng trường hợp sau:
  - ✓ Trường hợp 1:

```
set serveroutput on;
CREATE OR REPLACE TRIGGER TRG_TEST
AFTER UPDATE OF salary ON employee
FOR EACH ROW
DECLARE
   v_Count NUMBER;
BEGIN
   SELECT count(*)
   INTO v_count
   FROM employee
   WHERE salary =1000;
   dbms_output.put_line('Total employee are ' || v_count);
END;

update employee
set salary=1000
where salary=800;
```

Lưu ý: đoạn lệnh này có sử dụng câu lệnh **FOR EACH ROW**

✓ Trường hợp 2:

```
set serveroutput on;
CREATE OR REPLACE TRIGGER TRG_TEST
AFTER UPDATE OF salary ON employee
DECLARE
  v_Count NUMBER;
BEGIN
  SELECT count(*)
  INTO v_count
  FROM employee
  WHERE salary =1000;
  dbms_output.put_line('Total employee are ' || v_count);
END;

update employee
set salary=1000
where salary=800;
```

✓ Trường hợp 3:

```
CREATE OR REPLACE TRIGGER TRG_TEST
AFTER UPDATE OF salary ON employee
FOR EACH ROW
DECLARE
  PRAGMA AUTONOMOUS_TRANSACTION;
  v_Count NUMBER;
BEGIN
  SELECT count(*) INTO v_count
  FROM employee
  WHERE salary = 1000;
  dbms_output.put_line('Total employee are ' || v_count);
END;

update employee
set salary=1000
where salary =800;
```

Lưu ý: đoạn lệnh này có sử dụng câu lệnh **PRAGMA AUTONOMOUS_TRANSACTION;**

✓ Trường hợp 4:

```
CREATE OR REPLACE TRIGGER TRG_TEST
FOR UPDATE ON employee
COMPOUND TRIGGER
  /* Declaration Section*/
  v_count NUMBER;
  AFTER EACH ROW IS
  BEGIN
    dbms_output.put_line('Update is done');
  END AFTER EACH ROW;

  AFTER STATEMENT IS
  BEGIN
    SELECT count(*) INTO v_count
    FROM employee
    WHERE salary = 1000;
    dbms_output.put_line('Total employee are ' || v_count);
  END AFTER STATEMENT;
END TUA_TEST;

update employee
set salary=1000
where salary =800;
```

Lưu ý: đoạn lệnh này có sử dụng câu lệnh **COMPOUND TRIGGER;**

III. Bài tập làm tại lớp

Bài 1. Sử dụng CSDL Quản lý sinh viên (Buổi 2 / Bài 3 – phần bài tập mở rộng – về nhà), sinh viên hoàn thành các câu sau:

1) Viết các lệnh thực hiện những công việc sau:
   a. Tạo một bảng **Cau1** với 2 cột ID (number) và NAME (varchar2(20)).
   b. Tạo một sequence Cau1Seq với bước tăng là 5.
   c. Khai báo 2 biến **v_name và v_id**. Biến **v_name**, **v_id**. dùng để chứa giá trị họ, mã của **sinh viên** được thêm vào.
   d. Thêm vào bảng **Cau1** tên của sinh viên đã đăng kí trong các môn học (bảng **enrollment**) nhiều nhất. Mã sinh viên sẽ được lấy từ sequence Cau1Seq**.** Sau thao tác này tạo Savepoint A.
   e. Thêm vào bảng **Cau1** tên của sinh viên đã đăng kí trong các môn học (bảng **enrollment**) ít nhất. Mã sinh viên sẽ được lấy từ sequence Cau1Seq**.** Sau thao tác này tạo Savepoint B.
   f. Làm tương tự đối với các giáo viên có số lượng môn học dạy nhiều nhất. Sau thao tác này tạo Savepoint C.
   g. Sử dụng câu lệnh SELECT INTO, chứa giá trị của giáo viên có tên tương ứng v_name vào biến v_id.
   h. Undo giáo viên được thêm vào sử dụng rollback.

i. Thêm vào bảng **Cau1** giáo viên dạy ít môn học nhất nhưng mã thêm vào không lấy từ sequence mà lấy mã của giáo viên bị rollback trước đó.
j. Làm lại câu f với ID là lấy từ sequence.

2) Viết một đoạn chương trình: người dùng nhập vào mã sinh viên. Nếu sinh viên đó tồn tại thì hiển thị ra họ tên sinh viên và số lớp sinh viên đó đang học. Ngược lại, yêu cầu người dùng thêm vào sinh viên mới với mã số vừa nhập, các thông tin khác (họ, tên sinh viên, địa chỉ người dùng sẽ nhập vào).

Bài 2. Các cấu trúc điều khiển

1) Viết một đoạn mã lệnh: Người dùng nhập vào mã của một giáo viên, xác định số lượng lớp mà giáo viên này đang dạy. Nếu số lớp lớn hơn hoặc bằng 5 thì đưa ra một thông báo: "Giáo viên này nên nghỉ ngơi!", ngược lại in ra số lớp giáo viên này đang dạy.
2) Viết một đoạn mã lệnh (dùng cấu trúc case): Người dùng nhập vào mã của một sinh viên, mã lớp mà sinh viên này đang học. In ra điểm chữ của sinh viên này: A(90-100), B(80-90), C(70-80), D(50-70) F(0-50). Đồng thời in thông báo lỗi tương ứng khi người dùng nhập mã sinh viên hay mã lớp không tồn tại.

IV. Bài tập mở rộng – về nhà

Bài 3. Cursor

Viết một đoạn chương trình in ra thông tin các môn học và các lớp học thuộc môn học, số lượng sinh viên đăng kí lớp học như sau:

```
10 DP Overview
      Lop:  2 co so luong sinh vien dang ki: 1
20 Intro to Computers
      Lop:  2 co so luong sinh vien dang ki: 3
      Lop:  4 co so luong sinh vien dang ki: 2
 ....
```

Trong đó: **"20 Intro to Computers"** : 20 là mã môn học (courseno), Intro to Computers: là tên môn học (description); **"Lop: 2 co so luong sinh vien dang ki: 3"**: 2 là mã lớp học của môn tương ứng (classid), 3 là số lương sinh viên đăng kí lớp học này (count(*)).

Gợi ý: Tạo hai con trỏ (con trỏ sau có đối số là mã môn học), duyệt lần lượt 2 con trỏ này lồng nhau.

Bài 4. Các thủ tục và hàm

1) Viết 2 thủ tục
   a. Thủ tục find_sname có 1 thông số truyền vào (i_student_id), và 2 thông số trả về (o_first_name ,o_last_name) là họ và tên tương ứng của sinh viên với mã số truyền vào.
   b. Thủ tục print_student_name in ra tên của sinh viên với mã số là đối số truyền vào của thủ tục.
2) Viết thủ tục Discount giảm giá 5% cho tất cả các môn học có số sinh viên đăng kí nhiều hơn 15 sinh viên. Ứng với mỗi môn học được giảm giá in ra tên môn học đó.
3) Viết hàm Total_cost_for_student nhận vào mã số của sinh viên tra về tổng chi phí mà sinh viên đó phải trả. Trả về NULL nếu không tồn tại sinh viên tương ứng.

Bài 5. Trigger

1) Viết trigger cho các tác vụ thêm vào (insert), hay cập nhật (update) cho tất cả các bảng trong lược đồ quan hệ với các trường **created_by, created_date, modified_by, modified_date**, sẽ do trigger này thêm vào tương ứng với user hiện tại, ngày hệ thống hiện tại.
   Gợi ý: Dùng các hàm USER, SYSDATE để lấy được người dùng hiện tại, và ngày giờ của hệ thống.
2) Viết trigger hiện thực yêu cầu sau: mỗi sinh viên không được đăng kí quá 4 môn học.