* Oracle Database
* Tập hợp các file để lưu trữ và truy vấn dữ liệu, có cấu trúc vật lý và cấu trúc logic
* Cấu trúc vật lý gồm 3 loại file:

+ Data files: Chứa dữ liệu trong database

+ Online redo log files: Lưu tất cả các thay đổi trên CSDL khi instance được khởi động, cho phép khôi phục dữ liệu trong TH xảy ra lỗi

+ Control files: ghi lại cấu trúc vật lý của CSDL như tên của database

* Data files
* Chứa dữ liệu của database, bao gồm dữ liệu của user hay ứng dụng, data dictionary của oracle database
* SPFile
* Điều chỉnh các tham số trong SPFile

+ Cú pháp:

Alter system set parameter\_name = parameter\_value {memory|spfile|both}

* Tạo một SPFile từ Pfile
* **SQL>** CREATE SPFILE [='SPFILE-NAME']FROM PFILE[='PFILE-NAME'];
* Tạo Pfile từ SPFile
* **SQL>** CREATE PFILE [='PFILE-NAME']FROM SPFILE[=‘SPFILE-NAME'];
* Khởi động và tắt một oracle instance

1. Khởi động :

Để gõ lệnh **startup**, vào Sqlplus thông qua cmd, với quyền sysdba:

**Vào cmd, gõ: Sqlplus / as sysdba**

1. Thư mục mặc định của file tham số:

%Oracle\_home%\database

1. Khởi động CSDL

**STARTUP [**FORCE**][**RESTRICT**]** **[**PFILE=*filename***]** **[**MOUNT**]** **|** **[**OPEN **[***open\_options***]]** **|** **[**NOMOUNT**]**

1. **Thay đổi trạng thái của CSDL từ NOMOUNT sang MOUNT** hoặc từ MOUNT sang OPEN:
2. **Mở CSDL ở chế độ read-only: (chỉ thực hiện được khi database ở chế độ MOUNT)**
3. Câu lệnh SQL

Ngôn ngữ truy vấn có cấu trúc (SQL) là một ngôn ngữ lập trình được sử dụng để truy xuất, chèn, cập nhật và xóa dữ liệu từ cơ sở dữ liệu quan hệ. Các câu lệnh SQL thường dùng bao gồm:

* Câu lệnh SELECT được sử dụng để truy xuất dữ liệu từ cơ sở dữ liệu. Mệnh đề SELECT xác định các cột dữ liệu mà bạn muốn truy xuất, mệnh đề FROM xác định bảng mà bạn muốn truy xuất dữ liệu từ, và mệnh đề WHERE xác định các điều kiện lọc dữ liệu.

Ví dụ: câu lệnh sau sẽ truy xuất tất cả các cột từ bảng customers:

SQL

SELECT \*

FROM customers;

Vui lòng thận trọng khi sử dụng mã. [Tìm hiểu thêm](https://bard.google.com/faq#coding)

content\_copy

* Câu lệnh DISTINCT được sử dụng để loại bỏ các bản ghi trùng lặp từ kết quả truy vấn.

Ví dụ: câu lệnh sau sẽ truy xuất tất cả các khách hàng, nhưng chỉ một lần duy nhất, từ bảng customers:

SQL

SELECT DISTINCT customer\_id, customer\_name

FROM customers;

Vui lòng thận trọng khi sử dụng mã. [Tìm hiểu thêm](https://bard.google.com/faq#coding)

content\_copy

* Câu lệnh WHERE được sử dụng để lọc dữ liệu dựa trên các điều kiện.

Ví dụ: câu lệnh sau sẽ truy xuất tất cả các khách hàng có tên bắt đầu bằng chữ "A":

SQL

SELECT \*

FROM customers

WHERE customer\_name LIKE 'A%';

Vui lòng thận trọng khi sử dụng mã. [Tìm hiểu thêm](https://bard.google.com/faq#coding)

content\_copy

* Câu lệnh AND/OR được sử dụng để kết hợp các điều kiện lọc.

Ví dụ: câu lệnh sau sẽ truy xuất tất cả các khách hàng có tên bắt đầu bằng chữ "A" hoặc có địa chỉ ở thành phố "Hà Nội":

SQL

SELECT \*

FROM customers

WHERE customer\_name LIKE 'A%'

OR address = 'Hà Nội';

Vui lòng thận trọng khi sử dụng mã. [Tìm hiểu thêm](https://bard.google.com/faq#coding)

content\_copy

* Câu lệnh IN được sử dụng để lọc dữ liệu dựa trên một tập giá trị.

Ví dụ: câu lệnh sau sẽ truy xuất tất cả các khách hàng có tên trong tập giá trị ["Nguyễn Văn A", "Trần Thị B", "Lê Quang C"]:

SQL

SELECT \*

FROM customers

WHERE customer\_name IN ('Nguyễn Văn A', 'Trần Thị B', 'Lê Quang C');

Vui lòng thận trọng khi sử dụng mã. [Tìm hiểu thêm](https://bard.google.com/faq#coding)

content\_copy

* Câu lệnh ORDER BY được sử dụng để sắp xếp kết quả truy vấn theo một hoặc nhiều cột.

Ví dụ: câu lệnh sau sẽ sắp xếp kết quả truy vấn theo cột customer\_name theo thứ tự tăng dần:

SQL

SELECT \*

FROM customers

ORDER BY customer\_name;

Vui lòng thận trọng khi sử dụng mã. [Tìm hiểu thêm](https://bard.google.com/faq#coding)

content\_copy

* Câu lệnh GROUP BY được sử dụng để nhóm dữ liệu theo một hoặc nhiều cột và tính toán các số liệu thống kê.

Ví dụ: câu lệnh sau sẽ nhóm dữ liệu theo cột city và tính tổng số khách hàng theo từng thành phố:

SQL

SELECT city, COUNT(\*) AS total\_customers

FROM customers

GROUP BY city;

Vui lòng thận trọng khi sử dụng mã. [Tìm hiểu thêm](https://bard.google.com/faq#coding)

content\_copy

* Câu lệnh HAVING được sử dụng để lọc các nhóm dữ liệu dựa trên các điều kiện.

Ví dụ: câu lệnh sau sẽ nhóm dữ liệu theo cột city và chỉ hiển thị các nhóm có tổng số khách hàng lớn hơn 100:

SQL

SELECT city, COUNT(\*) AS total\_customers

FROM customers

GROUP BY city

HAVING total\_customers > 100;

Vui lòng thận trọng khi sử dụng mã. [Tìm hiểu thêm](https://bard.google.com/faq#coding)

content\_copy

* Câu lệnh JOIN được sử dụng để kết hợp dữ liệu từ hai hoặc nhiều bảng.

Có nhiều loại phép JOIN khác nhau, bao gồm:

\* \*\*INNER JOIN\*\* chỉ trả về các bản ghi có giá trị khớp trong cả hai bảng.

\* \*\*OUTER JOIN\*\* trả về tất cả các bản ghi từ bảng chính, ngay cả khi không có bản ghi khớp trong bảng liên kết.

\* \*\*CROSS JOIN\*\* trả về tất cả các cặp bản ghi có thể ghép từ hai bảng.

Ví dụ: câu lệnh sau sẽ kết hợp dữ liệu từ bảng customers và bảng orders để hiển thị tất cả các khách hàng đã đặt hàng:

SQL

SELECT customers.customer\_id, customers.customer\_name, orders.order\_id, orders.order\_date

FROM customers

INNER JOIN orders

ON customers.customer\_id = orders.customer\_id;

+ EXTRACT(YEAR | MONTH | DAY FROM <ngày> ): Trả về thành phần, ngày, tháng, hoặc năm của một dữ liệu kiểu date.

VD: SELECT EXTRACT(YEAR FROM TO\_DATE('3/3/2016','dd/mm/yyyy’)) FROM DUAL; (=>2016)

+ ADD\_MONTHS(*<ngày x>,<số tháng n>):* Trả về ngày mới sau khi cộng n tháng vào ngày x.

VD: SELECT add\_months(sysdate, 3) FROM DUAL;

+ MONTHS\_BETWEEN(*<ngày 1>,<ngày 2>):* Số tháng giữa 2 ngày.

+ SYSDATE: Trả về ngày tháng hiện tại.

1. Viết hàm đếm số nhân viên theo mã phòng ban:

sql

Copy code

CREATE FUNCTION dem\_nhanvien (p\_department\_id IN NUMBER)

RETURN NUMBER

IS

v\_count NUMBER;

BEGIN

SELECT COUNT(\*) INTO v\_count

FROM employees

WHERE department\_id = p\_department\_id;

RETURN v\_count;

END;

1. Viết thủ tục in ra thông tin nhân viên (employee\_id, name) theo mã phòng ban:

sql

Copy code

CREATE PROCEDURE in\_thongtin\_nhanvien (p\_department\_id IN NUMBER)

IS

v\_emp\_id NUMBER(6);

v\_name VARCHAR2(25);

BEGIN

FOR emp IN (SELECT employee\_id, first\_name||' '||last\_name name

FROM employees WHERE department\_id = p\_department\_id) LOOP

v\_emp\_id := emp.employee\_id;

v\_name := emp.name;

DBMS\_OUTPUT.PUT\_LINE(v\_emp\_id || ' - ' || v\_name);

END LOOP;

END;

1. Viết hàm tính tuổi nhân viên dựa vào ngày sinh:

sql

Copy code

CREATE FUNCTION tinh\_tuoi (p\_birthdate IN DATE)

RETURN NUMBER

IS

v\_age NUMBER(3);

BEGIN

SELECT FLOOR(MONTHS\_BETWEEN(SYSDATE, p\_birthdate)/12) INTO v\_age FROM dual;

RETURN v\_age;

END;

1. Viết hàm trả về mức lương trung bình của nhân viên theo mã phòng ban:

sql

Copy code

CREATE FUNCTION luong\_tb\_phong(p\_department\_id IN NUMBER)

RETURN NUMBER

IS

v\_avg\_sal NUMBER(8,2);

BEGIN

SELECT AVG(salary) INTO v\_avg\_sal

FROM employees

WHERE department\_id = p\_department\_id;

RETURN v\_avg\_sal;

END;

1. Viết thủ tục in ra thông tin địa chỉ của tất cả nhân viên ở 1 thành phố cụ thể:

sql

Copy code

CREATE PROCEDURE in\_diachi\_nhanvien (p\_city IN VARCHAR2)

IS

CURSOR cur\_emp IS

SELECT employee\_id, street\_address, postal\_code

FROM employees e JOIN departments d

ON e.department\_id = d.department\_id

JOIN locations l ON d.location\_id = l.location\_id

WHERE l.city = p\_city;

v\_emp\_id employees.employee\_id%TYPE;

v\_address employees.street\_address%TYPE;

v\_postal\_code locations.postal\_code%TYPE;

BEGIN

OPEN cur\_emp;

LOOP

FETCH cur\_emp INTO v\_emp\_id, v\_address, v\_postal\_code;

EXIT WHEN cur\_emp%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(v\_emp\_id || ' - ' || v\_address || ', ' || v\_postal\_code);

END LOOP;

CLOSE cur\_emp;

END;

1. Viết hàm trả về số lượng nhân viên theo từng job\_id:

sql

Copy code

CREATE FUNCTION dem\_nhanvien\_theo\_job(p\_job\_id IN VARCHAR2)

RETURN NUMBER IS

v\_count NUMBER;

BEGIN

SELECT COUNT(\*) INTO v\_count

FROM employees

WHERE job\_id = p\_job\_id

GROUP BY job\_id;

RETURN v\_count;

END;

1. Viết thủ tục cập nhật mức lương mới cho nhân viên theo phòng ban:

sql

Copy code

CREATE PROCEDURE update\_luong\_theo\_phong(

p\_department\_id IN NUMBER,

p\_luong\_moi IN NUMBER)

IS

BEGIN

UPDATE employees

SET salary = p\_luong\_moi

WHERE department\_id = p\_department\_id;

END;

1. Viết hàm kiểm tra xem một nhân viên có phải là quản lý hay không dựa vào employee\_id:

sql

Copy code

CREATE FUNCTION is\_manager(p\_emp\_id IN NUMBER)

RETURN VARCHAR2

IS

v\_result VARCHAR2(3);

BEGIN

SELECT DECODE(COUNT(\*), 0, 'No', 'Yes')

INTO v\_result

FROM employees

WHERE employee\_id = p\_emp\_id

AND manager\_id IS NULL;

RETURN v\_result;

END;

TẠO STORE PROCEDURE

43. Tạo thủ tục có tên là dept\_info cho biết thông tin về phòng ban với tham số truyền vào là mã phòng ban.

create or replace procedure dept\_info(v\_department\_id number, ten out departments.department\_name%type)

as begin

select department\_name into ten

from departments

where department\_id=v\_department\_id; dbms\_output.put\_line('Ten phong ban: '||ten);

exception when no\_data\_found

then dbms\_output.put\_line('Khong co phong ban');

end;

--Thuc thi

set serveroutput on

declare ten departments.department\_name%type; begin

DEPT\_INFO(&v\_department\_id, ten); end;

44. Tạo thủ tục có tên là add\_job thêm một công việc mới với tham số truyền vào là mã công việc, tên công việc.

create or replace procedure add\_job (v\_macv JOBS.job\_id%TYPE, v\_tencv JOBS.job\_title%TYPE)

as

v\_macv\_temp JOBS.job\_id%TYPE; v\_loi EXCEPTION;

begin

select job\_id into v\_macv\_temp

from JOBS

where job\_id=v\_macv; if v\_macv\_temp is not null then

raise v\_loi; end if;

exception when v\_loi then

dbms\_output.put\_line('Khong them duoc'); when no\_data\_found then

insert into JOBS (job\_id,job\_title) values (v\_macv,v\_tencv);

dbms\_output.put\_line('Cong viec ' || v\_tencv || ' da duoc

them.'); end;

--thuc thi

set serveroutput on

execute add\_job('p\_code','Lap Trinh Vien');

45. Tạo thủ tục có tên là update\_comm cập nhật hoa hồng cho nhân viên tang thêm 5% hoa hồng ban dầu, tham số truyền vào là mã nhân viên.

create or replace procedure update\_comm2(manhanvien number)

as begin

update employees

set commission\_pct = commission\_pct\*1.05

where employee\_id=manhanvien;

dbms\_output.put\_line('Cap nhat hoa hong cua '|| manhanvien ||'

thanh cong!'); end;

--thuc thi

set serveroutput on

execute update\_comm2(3);

46. Tạo thủ tục có tên là add\_emp thêm một nhân viên mới với tất cả các giá trị là tham số truyền vào.

create or replace procedure add\_emp(v\_employee\_id number

,first\_name varchar2, last\_name varchar2, email varchar2, phone\_number

varchar2,hire\_date employees.hire\_date%type, job\_id varchar2, salary number, commission\_pct number

, manager\_id number, department\_id employees.department\_id%type)

as

v\_count number; v\_loi exception;

begin

select count(\*) into v\_count from employees

where employee\_id=v\_employee\_id; if v\_count > 0 then raise v\_loi; else

insert into employees values(v\_employee\_id, first\_name, last\_name, email

, phone\_number, hire\_date, job\_id, salary, commission\_pct, manager\_id,

department\_id); dbms\_output.put\_line('Them thanh cong!'); end if;

exception when v\_loi then

dbms\_output.put\_line('Nhan vien da ton tai'); end;

47. Tạo thủ tục có tên là delete\_emp xóa một nhân viên mới với mã nhân

viên là tham số truyền vào.

create or replace procedure delete\_employee(manhanvien number)

as begin

delete from employees where employee\_id=manhanvien;

dbms\_output.put\_line('Da xoa'||manhanvien||'thanh cong!'); end;

48. Tạo thủ tục có tên find\_emp tìm kiếm nhân viên có lương lớn hơn

mức lương thấp nhất

create or replace procedure find\_emp

as

cursor c\_nhanvien is

select employee\_id, first\_name,last\_name

from EMPLOYEES a

where salary>(select min\_salary from jobs b where a.job\_id=b.job\_id) and salary<(select max\_salary from jobs c where a.job\_id=c.job\_id); v\_emp\_id EMPLOYEES.employee\_id%TYPE;

v\_first\_name EMPLOYEES.first\_name%TYPE; v\_last\_name EMPLOYEES.last\_name%TYPE;

begin

for r\_nhanvien in c\_nhanvien

loop

v\_emp\_id := r\_nhanvien.employee\_id;

v\_first\_name := r\_nhanvien.first\_name;

v\_last\_name:=r\_nhanvien.last\_name;

dbms\_output.put\_line('Ma NV: ' || v\_emp\_id || ' Ten

v\_first\_name || ' ' || v\_last\_name);

end loop;

end;

--Thuc thi

set serveroutput on

execute find\_emp;

49. Tạo thủ tục có tên update\_comm cập nhật luong của nhân

diều kiện nhân viên nào làm việc trên 2 nam thì tang luong

nhân viên làm việc trên 1 nam và duới 2 nam thì tang luong

nhân viên nào làm việc dúng 1 nam thì tang 50$, còn lại kh

create or replace procedure update\_comm

as

v\_sonam\_lamviec varchar2(2);

cursor c\_nhanvien is

select \* from EMPLOYEES;

begin

for r\_nhanvien in c\_nhanvien

loop

v\_sonam\_lamviec := to\_char(sysdate,'yyyy')-

to\_char(r\_nhanvien.hire\_date,'yyyy');

if v\_sonam\_lamviec>=2 then

update EMPLOYEES

set salary=salary+200

where employee\_id=r\_nhanvien.employee\_id;

dbms\_output.put\_line('Nhan vien :' ||

r\_nhanvien.last\_name || ' da duoc tang 200$');

else

update EMPLOYEES

set salary=salary+100

where employee\_id=r\_nhanvien.employee\_id;

dbms\_output.put\_line('Nhan vien :' ||

r\_nhanvien.last\_name || ' da duoc tang 100$');

end if;

end loop;

end;

--thuc thi

set serveroutput on

execute update\_comm;

TẠO FUNCTION

--51

create or replace function sum\_salary(maphongban in number)

return number

as

v\_salary number;

begin selectsum(salary) into v\_salary from employees

where department\_id=maphongban; return v\_salary;

exception

when no\_data\_found

then

return('Du lieu khong tim thay');

when others then return('loi ham'); end;

--Test

set serveroutput on

show error;

select column\_name, data\_type, data\_length from user\_tab\_columns

where table\_name='EMPLOYEES';

--

SELECT DEPARTMENT\_ID FROM DEPARTMENTS;

set serveroutput on

set verify off

execute dbms\_output.put\_line('Tong luong la: '||sum\_salary(&maphong));

--52.Tao ham co ten name\_con cho biet ten quoc gia voi ma quoc gia la

tham so --truyen vao

--Xem kieu du lieu

select column\_name, data\_type, data\_length from user\_tab\_columns

where table\_name='COUNTRIES';

--52

createorreplacefunction name\_con (maqg in countries.country\_id%type) return varchar2

is

tenqg varchar2(50);

begin

select country\_name into tenqg

from countries

where country\_id=maqg; return tenqg;

exception

when no\_data\_found

then

return('Du lieu khong tim thay');

when others then return('loi ham'); end;

--Thuc thi

SELECT COUNTRY\_ID FROM COUNTRIES;

--Goi thuc thi 1

set serveroutput on set verify off

declare quocgia varchar2(35); begin

quocgia:=name\_con('AU');

dbms\_output.put\_line(quocgia); end;

--Goi thuc thi 2 --53

createorreplacefunction annual\_comp (luong employees.salary%type,

hoahong employees.commission\_pct%type) return number

as

thunhap number;

begin thunhap:=luong\*12+(hoahong\*luong\*12);

return thunhap;

exception

when no\_data\_found

then

return('Du lieu khong tim thay');

when others then return('loi ham'); end;

--Goi ham

select salary, commission\_pct from EMPLOYEES;

--

set serveroutput on set verify off

declare ThuNhap number; begin

ThuNhap:=annual\_comp(14200,0.4);

dbms\_output.put\_line(ThuNhap); end;

--54

Create or replace function avg\_salary (mapb employees.department\_id%type) return number

as

luongtb number;

begin selectavg(salary) into luongtb from employees

where department\_id=mapb; return luongtb;

exception

when no\_data\_found

then

return('Du lieu khong tim thay');

when others then return('loi ham'); end;

--Goi thuc thi

set serveroutput on; set verify off

declare LuongTB number; begin

LuongTB:=avg\_salary(110);

dbms\_output.put\_line(LuongTB); end;

--55: thời gian làm việc của từng nhân viên

Create or replace function Time\_work(MaNhanVien EMPLOYEES.EMPLOYEE\_ID%TYPE)

return number

as

tglamviec number;

begin

select round(months\_between(to\_date(sysdate,'dd/mm/yyyy'), to\_date(hire\_date,'dd/mm/yyyy')),1)

into tglamviec

from EMPLOYEES

where employee\_id=MaNhanVien; return tglamviec;

exception when no\_data\_found then return('Du lieu khong tim thay'); end;

--Goi thuc thi

set serveroutput on declare tg number(22); begin tg:=Time\_work(&manv);

dbms\_output.put\_line('So thang lam viec cua nhan vien la '||tg); end;

show error

--TEST

select\*from employees;

--xem kieu du lieu cua cot

select column\_name,data\_type, data\_length

from user\_tab\_columns

where table\_name='EMPLOYEES'

select round(months\_between (to\_date(sysdate,'dd/mm/yyyy'),to\_date(hire\_date,'dd/mm/yyyy')),2) from employees

create user htn identified by abc123;

grant create table to htn;

grant create session to htn;

connect htn/abc123;

-- T?o b?ng TBLLoaiSach

CREATE TABLE TBLLoaiSach (

maLoai VARCHAR2(10) NOT NULL,

tenLoai NVARCHAR2(100) NOT NULL,

ghiChu NVARCHAR2(1000),

PRIMARY KEY (maLoai)

);

-- T?o b?ng TBLNXB

CREATE TABLE TBLNXB (

maNXB VARCHAR2(10) NOT NULL,

tenNXB NVARCHAR2(100) NOT NULL,

diaChi NVARCHAR2(200),

PRIMARY KEY (maNXB)

);

-- T?o b?ng TBLSach

CREATE TABLE TBLSach (

maSach NUMBER NOT NULL,

tenSach NVARCHAR2(500) NOT NULL,

maLoai VARCHAR2(10) NOT NULL,

maNXB VARCHAR2(10) NOT NULL,

tacGia NVARCHAR2(50),

namXB NUMBER,

soLuong NUMBER,

PRIMARY KEY (maSach),

FOREIGN KEY (maLoai) REFERENCES TBLLoaiSach (maLoai),

FOREIGN KEY (maNXB) REFERENCES TBLNXB (maNXB)

);

-- T?o b?ng TBLNhanVien

CREATE TABLE TBLNhanVien (

maNV NUMBER NOT NULL,

username VARCHAR2(100) NOT NULL,

password VARCHAR2(100) NOT NULL,

gioitinh VARCHAR2(3),

PRIMARY KEY (maNV)

);

CREATE OR REPLACE PROCEDURE find\_book (

p\_book\_name VARCHAR2,

p\_author VARCHAR2,

p\_category VARCHAR2

)

IS

v\_book\_id NUMBER;

BEGIN

SELECT s.maSach

INTO v\_book\_id

FROM TBLSach s

JOIN TBLLoaiSach l ON s.maLoai = l.maLoai

WHERE s.tenSach = p\_book\_name

AND s.tacGia = p\_author

AND l.tenLoai = p\_category;

DBMS\_OUTPUT.PUT\_LINE('Mã sách tìm ???c: ' || v\_book\_id);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Không tìm th?y sách phù h?p');

END;

/

BEGIN

find\_book('K? thu?t l?p trình C', 'Ph?m V?n ?t', 'K? thu?t');

END;

/

set serveroutput on;

declare

message varchar2(20):='Hello';

begin

dbms\_output.put\_line(message);

end;

disconnect

connect system/17012003;

CREATE OR REPLACE PROCEDURE find\_book (

p\_book\_name VARCHAR2,

p\_author VARCHAR2,

p\_category VARCHAR2

)

IS

v\_book\_id NUMBER;

BEGIN

SELECT s.maSach

INTO v\_book\_id

FROM TBLSach s

JOIN TBLLoaiSach l ON s.maLoai = l.maLoai

WHERE s.tenSach = p\_book\_name

AND s.tacGia = p\_author

AND l.tenLoai = p\_category;

DBMS\_OUTPUT.PUT\_LINE('Mã sách tìm ???c: ' || v\_book\_id);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Không tìm th?y sách phù h?p');

END;

/

set serveroutput on;

BEGIN

find\_book('K? thu?t l?p trình C', 'Ph?m V?n ?t', 'K? thu?t');

END;

/

SELECT PROCEDURE\_NAME

FROM ALL\_PROCEDURES;

grant execute on find\_book to hr;

connect hr/17012003;

CREATE FUNCTION tong\_luong\_phong (p\_department\_id NUMBER)

RETURN NUMBER

IS

v\_tong\_luong NUMBER(10,2);

BEGIN

SELECT SUM(e.salary)

INTO v\_tong\_luong

FROM employees e

WHERE e.department\_id = p\_department\_id

GROUP BY e.department\_id;

RETURN v\_tong\_luong;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 0;

END;

/

set serveroutput on;

CREATE OR REPLACE PROCEDURE thong\_ke\_luong\_phong

IS

CURSOR c\_dept IS

SELECT department\_id, department\_name FROM departments;

v\_dept\_id departments.department\_id%TYPE;

v\_dept\_name departments.department\_name%TYPE;

v\_tong\_luong NUMBER(10,2);

BEGIN

OPEN c\_dept;

LOOP

FETCH c\_dept INTO v\_dept\_id, v\_dept\_name;

EXIT WHEN c\_dept%NOTFOUND;

v\_tong\_luong := tong\_luong\_phong(v\_dept\_id);

DBMS\_OUTPUT.PUT\_LINE('Phòng ' || v\_dept\_name || ': ' || v\_tong\_luong);

END LOOP;

CLOSE c\_dept;

END;

/

set serveroutput on;

begin

thong\_ke\_luong\_phong;

end;