**Apartat 1. Procediments.**

**Exercici 1.**

Fes un procediment emmagatzemat que rebi com a paràmetres les dades d’un departament i el doni d’alta.

create or replace function f\_hola(dep\_id departments.department\_id%type,

dep\_name departments.department\_name%type,

dep\_loc departments.location\_id%type)

returns void

language plpgsql as

$$

begin

insert into departments(department\_id, department\_name, location\_id)

values(dep\_id, dep\_name, dep\_loc);

end $$;

-- Procediment

create or replace procedure p\_hola(dep\_id departments.department\_id%type,

dep\_name departments.department\_name%type,

dep\_loc departments.location\_id%type)

language plpgsql as

$$

declare

n int;

begin

insert into departments(department\_id, department\_name, location\_id)

values(dep\_id, dep\_name, dep\_loc);

end $$;

do $$

declare

n\_dep departments.department\_id%type;

begin

select max(department\_id)+1 into n\_dep from departments;

-- Perform permet executar sentències sql descartant els resultats, no cal la clàusula Into

perform f\_hola(n\_dep, 'Nou dep', 1500);

call p\_hola(n\_dep+1, ‘Nou dep + 1’, 1400);

end $$

**Exercici 2.**

Fes un procediment emmagatzemat que donat un codi de departament, l’elimini. En cas que es produeixi un error al eliminar-lo que avisi per consola amb el següent missatge:

Error: No es pot eliminar el departament indicat.

CREATE OR REPLACE FUNCTION DeleteDepartment(dept\_id DEPARTMENTS.DEPARTMENT\_ID%TYPE) RETURNS void

LANGUAGE plpgsql AS $$

BEGIN

IF COALESCE(dept\_id,0) <> 0 THEN

DELETE FROM DEPARTMENTS WHERE DEPARTMENT\_ID = dept\_id;

END IF;

EXCEPTION

WHEN OTHERS THEN

raise exception 'No es pot eliminar el departament indicat';

END;$$;

do $$

DECLARE

dept\_id DEPARTMENTS.DEPARTMENT\_ID%TYPE:= 170;

x char;

begin

select DELETEDEPARTMENT(dept\_id) into x;

end;$$;

**Exercici 3.**

Fes un procediment emmagatzemat que li passis un codi d’empleat i un salari, i li assigni a aquell empleat el nou salari indicat. En cas de no trobar l’empleat que avisi amb un missatge.

CREATE OR REPLACE FUNCTION AssignSalary(empl\_id EMPLOYEES.EMPLOYEE\_ID%TYPE,

sal EMPLOYEES.SALARY%TYPE) RETURNS void

LANGUAGE plpgsql AS $$

DECLARE

N int;

BEGIN

IF coalesce(empl\_id,0) <> 0 THEN

UPDATE EMPLOYEES SET SALARY = sal WHERE EMPLOYEE\_ID = empl\_id;

GET DIAGNOSTICS N = ROW\_COUNT;

IF N = 0 THEN

RAISE 'Error actualització'

USING ERRCODE = 99999;

END IF;

END IF;

EXCEPTION

WHEN SQLSTATE '99999' THEN

RAISE 'No ha estat possible actualitzar el registre.';

END;$$;

select ASSIGNSALARY(104, 12000);

**Apartat 2. Funcions.**

**Exercici 4.**

Programar una funció que rebi com a paràmetre una nota (suposarem que l’entrada serà correcte entre 0 i 10) i retorni la seva nota qualitativa:

* Insuficient: de 0 a 4,99
* Suficient: de 5 a 5,99
* Bé: de 6 a 6,99
* Notable: de 7 a 8,99
* Excel·lent: de 9 a 9,99
* Matrícula d'honor: 10

CREATE OR REPLACE FUNCTION NotaFinal(nota NUMERIC) RETURNS VARCHAR

LANGUAGE plpgsql AS $$

DECLARE

resultat VARCHAR(50);

BEGIN

CASE

WHEN nota <5 THEN

resultat:='Insuficient';

WHEN nota >=5 AND nota < 6 THEN

resultat:='Suficient';

WHEN nota >=6 AND nota < 7 THEN

resultat:='Be';

WHEN nota >=7 AND nota < 9 THEN

resultat:='Notable';

WHEN nota >=9 AND nota < 10 THEN

resultat:= 'Excel.lent';

WHEN nota = 10 THEN

resultat:='Matricula d''honor';

ELSE

resultat:='Error: La nota ha d''estar entre 0 i 10';

END CASE;

RETURN resultat;

END;$$;

do $$

BEGIN

raise notice 'Nota: %', NOTAFINAL(6);

END;$$;