LABORATORIO IX

1.

```
CREATE OR REPLACE FUNCTION Verificacionxpagos()
RETURNS TRIGGER AS $$
DECLARE
TMPsalario INTEGER;
TMPbonificacion INTEGER;
BEGIN
IF NEW.codconcepto = 1
THEN
        IF NEW.valor =  (SELECT salario INTO TMPsalario FROM empsucursal NATURAL
        JOIN pagonomina WHERE pagonomina.nrocomprobante = NEW.nrocomprobante)

        THEN
        RETURN NEW;
        ELSE
        NEW.valor=TMPsalario;
        RETURN NEW;
        END IF;
END IF;
IF NEW.codconcepto = 2
THEN
        IF NEW.valor = (SELECT (salario*bonificacion) INTO TMPbonificacion FROM
        empsucursal  NATURAL JOIN pagonomina WHERE
        pagonomina.nrocomprobante=NEW.nrocomprobante)
        THEN
        RETURN NEW;
        ELSE
        NEW.valor=TMPbonificacion;
        RETURN NEW;
```

```
                END IF;


END IF;

END; $$ LANGUAGE 'plpgsql';


CREATE TRIGGER Verificacionxpagos BEFORE INSERT OR UPDATE ON detallepago FOR EACH ROW
EXECUTE PROCEDURE Verificacionxpagos();
```

2.

```
ALTER TABLE sucursal ADD COLUMN totalsalarios INTEGER;


SELECT codsucursal,SUM(salario) as totsalarios FROM empleado INNER JOIN empsucursal ON
empleado.id=empsucursal.idempleado GROUP BY codsucursal ORDER BY codsucursal;


UPDATE sucursal SET totalsalarios=647086317 WHERE codsucursal=100;

UPDATE sucursal SET totalsalarios=811777183 WHERE codsucursal=200;

UPDATE sucursal SET totalsalarios=642650185 WHERE codsucursal=300;

UPDATE sucursal SET totalsalarios=718617873 WHERE codsucursal=400;

UPDATE sucursal SET totalsalarios=667474630 WHERE codsucursal=500;

UPDATE sucursal SET totalsalarios=684580823 WHERE codsucursal=600;

UPDATE sucursal SET totalsalarios=777466955 WHERE codsucursal=700;

UPDATE sucursal SET totalsalarios=606365290 WHERE codsucursal=800;


CREATE OR REPLACE FUNCTION updateTotalSal()

RETURNS TRIGGER AS $$

DECLARE

tmptotsal INTEGER;

BEGIN
```

```
SELECT SUM(salario) INTO tmptotsal FROM empleado INNER JOIN empsucursal ON
empleado.id=empsucursal.idempleado WHERE codsucursal= OLD.codsucursal ;

UPDATE sucursal SET totalsalarios=tmptotsal WHERE codsucursal=OLD.codsucursal;

RETURN NULL;

END; $$ LANGUAGE 'plpgsql';


CREATE TRIGGER updateTotalSal AFTER INSERT OR UPDATE ON empsucursal FOR EACH ROW
EXECUTE PROCEDURE updateTotalSal();


UPDATE empsucursal SET salario=1000000 WHERE idempleado=1;


SELECT id,codsucursal FROM empleado INNER JOIN empsucursal ON
empleado.id=empsucursal.idempleado WHERE id=1;


SELECT * from sucursal;


UPDATE empsucursal SET salario=2000000 WHERE idempleado=1;


SELECT * from sucursal;


INSERT INTO empsucursal (codsucursal,idempleado,codcargo,salario) VALUES
(600,1500,1,1000000);


SELECT * from sucursal;
```

3.

```
CREATE OR REPLACE FUNCTION cadenasempleadopagos(inp_id INTEGER)

RETURNS VARCHAR AS $$

DECLARE

fila RECORD;

fila2 RECORD;
```

```
curs1 CURSOR IS SELECT empleado.nombre,empleado.apellido,ciudad.nombreciudad FROM
empleado NATURAL JOIN ciudad WHERE empleado.id=inp_id;

curs2 CURSOR IS SELECT fechapago,valor FROM empleado INNER JOIN pagonomina ON
empleado.id=pagonomina.idempleado NATURAL JOIN detallepago NATURAL JOIN conceptopago
WHERE empleado.id=inp_id and conceptopago.descripcion='Salario';

Vartext1 VARCHAR :='';

Vartext2 VARCHAR :='';

Vartext VARCHAR :='';


BEGIN

        FOR fila in curs1 LOOP

        Vartext1:='Empleado: '|| fila.nombre ||' '|| fila.apellido ||', ' || fila.nombreciudad ||
        E'\n';

        END LOOP;

        FOR fila2 in curs2 LOOP

        Vartext2:= Vartext2|| 'Pago: ' || fila2.fechapago || '; ' || fila2.valor || E'\n';

        END LOOP;

        Vartext:= Vartext1 || Vartext2;

        RETURN Vartext;

END; $$ LANGUAGE 'plpgsql';


SELECT * FROM cadenasempleadopagos(1);
```