

# Practica-3-Resuelta.pdf



**PedroChota**



**GESTIÓN Y ADMINISTRACIÓN DE BASES DE DATOS**



**3º Grado en Ingeniería Informática**



**Escuela Politécnica Superior (Jaén)  
Universidad de Jaén**

Formamos  
**talento** para un futuro  
**Sostenible**



MÁSTER EN

**Big Data &  
Business Analytics**

**EOI** Escuela de  
organización  
industrial

[saber más](#)

# ESTUDIAR QUEMA, PERO **FLAMIN' HOT** QUEMA MÁS

Wuolah: PedroChota

## Práctica 3 - Resuelta

OPERATION\_ID.sql:

```
1 CREATE SEQUENCE secuencia_id_op START WITH 23 MINVALUE 1 MAXVALUE
2 999 INCREMENT BY 1;
3 CREATE OR REPLACE TRIGGER actualizaIdOperation
4 BEFORE INSERT
5 ON Operation
6 FOR EACH ROW
7 BEGIN
8 :new.cod_op := secuencia_id_op.NEXTVAL;
9 END;
10 /
```

SALDO\_FI.sql:

```
1 DROP TABLE saldo_fi;
2
3 CREATE TABLE saldo_fi (
4 cod_fi SMALLINT NOT NULL CONSTRAINT pk_saldo_fi PRIMARY KEY,
5 saldo NUMBER (10,2) CONSTRAINT nn_saldo_saldofi NOT NULL);
6
7 CREATE OR REPLACE PROCEDURE carga_saldo_fi IS
8 CURSOR fondos IS
9 SELECT * FROM Fund;
10 CURSOR inversiones IS
11 SELECT * FROM OP_OSRFI;
12 BEGIN
13 /*Primero inicializamos cada fondo a 0*/
14 FOR fondo IN fondos LOOP
15 INSERT INTO saldo_fi VALUES(fondo.id,0);
16 END LOOP;
17 /*Actualizamos los fondos de cada fundación*/
18 FOR inversion IN inversiones LOOP
19 IF(inversion.operation='subscription') THEN
20 UPDATE saldo_fi s SET s.saldo=s.saldo+inversion.amount
21 WHERE inversion.Fund_id=s.cod_fi;
22 END IF;
23 IF(inversion.operation='refund') THEN
24 UPDATE saldo_fi s SET s.saldo=s.saldo-inversion.amount
25 WHERE inversion.Fund_id=s.cod_fi;
26 END IF;
27 END LOOP;
28 END;
29 /
30
31 EXECUTE carga_saldo_fi; /*Ejecución del procedimiento*/
32
33 SELECT * FROM saldo_fi;
```

WUOLAH

```

34 SELECT * FROM OP_OSRFI;
35 SELECT * FROM Fund;

```

### TGR\_DESCUENTO.sql:

```

1  CREATE OR REPLACE TRIGGER tgr_descubierto
2  BEFORE UPDATE OF balance ON account
3  FOR EACH ROW
4  DECLARE
5      fondos_retirados NUMBER;
6  BEGIN
7      IF (:new.balance < :old.balance) THEN
8          fondos_retirados := :old.balance - :new.balance;
9          IF (fondos_retirados > :old.balance * 1.10) THEN
10             RAISE_APPLICATION_ERROR(-20000, 'No se puede retirar un
11 saldo superior al 110% del saldo actual');
12             END IF;
13             END IF;
14 END;
15 SELECT * FROM account;

```

### TGR\_OP\_I.sql:

```

1  CREATE OR REPLACE TRIGGER TRG_OP_I
2  BEFORE INSERT ON op_i
3  FOR EACH ROW
4  DECLARE
5      dinero op_i.amount%type;
6      codigo op_i.cod_op%type;
7  BEGIN
8      dinero := :new.amount;
9      codigo := :new.cod_op;
10     UPDATE account ac SET ac.balance=ac.balance+dinero
11     WHERE ac.id = (SELECT o."Cli-Acc_id"
12                     FROM Operation o
13                     WHERE o.cod_op = codigo);
14 END;
15 /
16
17 SELECT * FROM account;
18 SELECT * FROM operation;
19 /*Prueba de trigger:*/
20 INSERT INTO Operation values (21, TO_DATE('2015-01-30 14:07:01',
21 'yyyy-mm-dd hh24:mi:ss'), 4);
22 INSERT INTO OP_I VALUES (21, 0049, 1653, 000676, 450);

```

### TGR\_SALDO\_FI.sql:

```

1  CREATE OR REPLACE TRIGGER TRG_SALDO_FI

```

```
2 AFTER INSERT
3   ON OP_OSRFI
4 FOR EACH ROW
5 DECLARE
6   dinero OP_OSRFI.amount%type;
7 BEGIN
8   IF(:new.operation='subscription') THEN
9     dinero := :new.amount;
10  END IF;
11  IF(:new.operation='refund') THEN
12    dinero := :new.amount;
13  END IF;
14
15  UPDATE saldo_fi s SET s.saldo=s.saldo+dinero
16  WHERE s.cod_fi = :new.Fund_id;
17 END;
18 /
19
20 SELECT * FROM Operation;
```