max_2_prov_pieza.sql

```
/* @op_BD_piezas_2.sql */
SET ECHO ON

DROP TRIGGER tst_maxProv_TR;

DELETE FROM suministrar WHERE (clvProv, clvPieza) IN ((2,94), (2,95));

SELECT * FROM Suministrar;

/* Crear un disparador para impedir que el nº de Proveedores por pieza sea > 2
*/
CREATE OR REPLACE TRIGGER tst_maxProv_TR
BEFORE INSERT ON Suministrar
FOR EACH ROW
DECLARE
  totProv number;
BEGIN
  SELECT count(*) INTO totProv
  FROM Suministrar
  WHERE :NEW.clvPieza = clvPieza;
  DBMS_OUTPUT.PUT_LINE('total proveedores de pieza '||'  '||totProv);

  IF totProv > 1 THEN
    raise_application_error( -20501, 'la pieza tiene el máximo de proveedores');
  END IF;
END tst_maxProv_TR;
/
SHOW ERRORS TRIGGER tst_maxProv_TR

-- no se producirá error, pues suministrar no es mutante, ¡para este caso particular!
INSERT INTO suministrar VALUES (2, 95);

-- SI se producirá error, pues suministrar es mutante
INSERT INTO suministrar SELECT 2, clvPieza FROM Suministrar WHERE clvPieza=93;
INSERT INTO suministrar SELECT 2, clvPieza FROM Suministrar WHERE clvPieza=92;

/* Crear un disparador para impedir que el nº de Proveedores por pieza sea > 2
*/
CREATE OR REPLACE TRIGGER tst_maxProv_TR
AFTER INSERT ON Suministrar
FOR EACH ROW
DECLARE
  totProv number;
BEGIN
  SELECT count(*) INTO totProv
  FROM Suministrar
  WHERE :NEW.clvPieza = clvPieza;
  DBMS_OUTPUT.PUT_LINE('total proveedores de pieza '||'  '||totProv);

  IF totProv > 1 THEN
    raise_application_error( -20501, 'la pieza tiene el máximo de proveedores');
  END IF;
END tst_maxProv_TR;
/
SHOW ERRORS TRIGGER tst_maxProv_TR

-- se producirá error, pues suministrar es mutante dentro del disparador
INSERT INTO suministrar VALUES (2, 95);
```

```
ALTER TABLE Pieza ADD(numProv number);

--ALTER TABLE Pieza DROP(numProv);


/* Crear un procedimiento para actualizar el número de proveedores de cada pieza
*/
CREATE OR REPLACE PROCEDURE actualizarSuministros_PR
IS
BEGIN
  UPDATE Pieza P SET numProv=(SELECT count(*) FROM Suministrar WHERE clvPieza = P.clvPieza);
END actualizarSuministros_PR;
/
SHOW ERRORS PROCEDURE actualizarSuministros_PR

EXECUTE actualizarSuministros_PR

SET linesize 132
SET pagesize 300
column color   format A32
column numProv format 999

SELECT * FROM Pieza;

DROP TRIGGER tst_maxProv_TR;

/* Crear un disparador para impedir que el nº de Proveedores por pieza sea > 3
*/
CREATE OR REPLACE TRIGGER tst_maxProv_TR
BEFORE INSERT ON Suministrar
FOR EACH ROW
DECLARE
  totProv number;
BEGIN
  SELECT numProv INTO totProv
  FROM Pieza
  WHERE :NEW.clvPieza = clvPieza;
  DBMS_OUTPUT.PUT_LINE('total proveedores de pieza '||'  '||totProv);

  IF totProv > 2
    THEN
      raise_application_error( -20501, 'la pieza tiene el máximo de proveedores');
    ELSE
      UPDATE Pieza SET numProv=numProv+1 WHERE clvPieza = :NEW.clvPieza; -- mejorable
  END IF;
END tst_maxProv_TR;
/

SHOW ERRORS TRIGGER tst_maxProv_TR

INSERT INTO suministrar VALUES (2, 91);
INSERT INTO suministrar VALUES (3, 92);
INSERT INTO suministrar VALUES (3, 91);
```

```
CREATE TABLE tempSum (
  clvProv  number(9),
  clvPieza number(9)
);

BEGIN NULL; END;
/

DROP TRIGGER tst_maxProv_TR;

/* Crear un disparador para impedir que el nº de Proveedores por pieza sea > 3
   Como no se puede consultar la tabla que se está modificando, se añaden los cambios
   a la tabla temporal tempSum y se crea un nuevo trigger que se active al terminar de
   modificar dicha tabla
*/
CREATE OR REPLACE TRIGGER tst_maxProv_TR
BEFORE INSERT ON Suministrar
FOR EACH ROW
BEGIN
  INSERT INTO tempSum VALUES(:NEW.clvProv, :NEW.clvPieza);
  DBMS_OUTPUT.PUT_LINE('intento de añadir suministro ' || :NEW.clvProv || ' ' || :NEW.clvPieza);
END tst_maxProv_TR;
/
SHOW ERRORS TRIGGER tst_maxProv_TR

DROP TRIGGER nuevosSuministros_TR;

CREATE OR REPLACE TRIGGER nuevosSuministros_TR
AFTER INSERT OR UPDATE ON Suministrar
DECLARE
  laPieza    Suministrar.clvPieza%TYPE;
  elProveedor Suministrar.clvProv%TYPE;
  totalProv   number;
  CURSOR selCambio IS SELECT clvPieza, clvProv FROM tempSum;
BEGIN
  OPEN selCambio;
  LOOP
    FETCH selCambio INTO laPieza, elProveedor;
    EXIT WHEN selCambio%NOTFOUND;
    SELECT count(*) INTO totalProv
      FROM Suministrar S
      WHERE S.clvPieza = laPieza;
    IF totalProv > 2 THEN
      DBMS_OUTPUT.PUT_LINE('la pieza '|| laPieza || ' ya tiene '|| (totalProv-1) || ' proveedores');
      raise_application_error( -20501, 'numero máximo de proveedores alcanzado');
    END IF;
  END LOOP;
  DELETE FROM tempSum;
END tst_maxProv_TR;
/
SHOW ERRORS TRIGGER nuevosSuministros_TR

DELETE FROM suministrar WHERE (clvProv, clvPieza) IN ((2,91), (3,92), (3,91));

SELECT * FROM Suministrar;

-- se producirá error, pues suministrar es mutante dentro del disparador
INSERT INTO suministrar VALUES (2, 91);
INSERT INTO suministrar VALUES (3, 92);
INSERT INTO suministrar VALUES (3, 91);



DROP TABLE tempSum;
```

```
/* Crear un disparador para impedir la modificación del nombre de una pieza
*/
CREATE OR REPLACE TRIGGER updateNombPieza_TR
BEFORE UPDATE ON Pieza
FOR EACH ROW
DECLARE
  laPieza number;
BEGIN
  laPieza := :OLD.clvPieza;
  IF UPDATING('NOMBPIEZA')
  THEN
    DBMS_OUTPUT.PUT_LINE('no se permite cambiar el nombre de una pieza');
    raise_application_error( -20501, 'error al modificar la pieza '|| :OLD.clvPieza);
  END IF;
END updateNombPieza_TR;
/
SHOW ERRORS TRIGGER updateNombPieza_TR

SELECT * FROM Pieza;

UPDATE Pieza SET color = 'ROJIZO' WHERE color='ROJO';

UPDATE Pieza SET nombPieza = 'FLEJE'  WHERE clvPieza=91;
UPDATE Pieza SET nombPieza = 'TUERCA' WHERE clvPieza=91;

UPDATE Pieza SET color = 'ROJO' WHERE color='ROJIZO';
--
--
/* Crear un disparador para impedir la modificación del nombre de una pieza
*/
CREATE OR REPLACE TRIGGER updateNombPieza_TR
BEFORE UPDATE OF nombPieza ON Pieza
FOR EACH ROW
WHEN (OLD.nombPieza <> NEW.nombPieza)
BEGIN
   raise_application_error( -20501, 'error al modificar la pieza '|| :OLD.clvPieza);
END updateNombPieza_TR;
/
SHOW ERRORS TRIGGER updateNombPieza_TR

SELECT * FROM Pieza;
UPDATE Pieza SET color = 'ROJIZO' WHERE color='ROJO';
UPDATE Pieza SET nombPieza = 'FLEJE'  WHERE clvPieza=91;
UPDATE Pieza SET nombPieza = 'TUERCA' WHERE clvPieza=91;
--
--
--DROP TRIGGER updateNombPieza_TR;


/* Crear un disparador para detectar las operaciones de creación y eliminación de elementos en el
esquema de la BD
*/
CREATE OR REPLACE TRIGGER Santiago
AFTER CREATE OR DROP ON SCHEMA
BEGIN
  IF ORA_SYSEVENT = 'CREATE'
  THEN
    DBMS_OUTPUT.PUT_LINE(ORA_DICT_OBJ_TYPE || ' de nombre ' || ORA_DICT_OBJ_NAME || ' se ha creado');
  ELSIF ORA_SYSEVENT = 'DROP'
  THEN
    DBMS_OUTPUT.PUT_LINE(ORA_DICT_OBJ_TYPE || ' de nombre ' || ORA_DICT_OBJ_NAME || ' se ha
eliminado' );
  END IF;
END;
/
```