```
USE IISSI_Friends;
-- trigger maxFotos usuarios
DELIMITER //
CREATE OR REPLACE TRIGGER maxFotosUsuario
BEFORE INSERT ON fotos
FOR EACH ROW
BEGIN
DECLARE n INT;
SET n = (SELECT COUNT(*) FROM fotos WHERE (email = new.email));
IF (n >= 5) THEN
 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =' `Un usuario no puede tener
mas de 5 fotos`';
END IF;
END //
DELIMITER ;

DELIMITER //
CREATE OR REPLACE TRIGGER vinAmistadDiarios
 BEFORE INSERT ON vinculos_amistades
FOR EACH ROW
BEGIN
DECLARE n INT;
SET n  = (SELECT COUNT(*) FROM vinculos_amistades WHERE (email1 =
NEW.email1 AND fecha_solicitud = CURDATE()));
IF (n> 5) then
SIGNAL SQLSTATE'45000' SET MESSAGE_TEXT = 'un usauario no puede lanzar mas
de 5 solicitudes por dia natural';
END if ;
END //
DELIMITER ;

DELIMITER //
CREATE OR REPLACE TRIGGER AutoExcusionVINAmistad BEFORE INSERT ON
vinculos_amistades
FOR EACH ROW
BEGIN
if(NEW.email1  = NEW.email2) then
SIGNAL SQLSTATE'45000' SET MESSAGE_TEXT = 'un usuario no puede tener un
vinculo consigo mismo';
END IF;
END //
DELIMITER ;
-- realizados por imane Aalouane

DELIMITER //
CREATE OR REPLACE TRIGGER  vinculoactivo
BEFORE INSERT ON vinculos_amistades
FOR EACH ROW
BEGIN
DECLARE n INT;
DECLARE d INT;
DECLARE g INT;
```

```
SET n = (SELECT COUNT(*) FROM vinculos_amistades v WHERE v.email1 =
NEW.email2 AND v.email2 = NEW.email1 AND v.VinculoActivo = TRUE);
SET d = (SELECT COUNT(*) FROM vinculos_amistades v WHERE v.email1 =
NEW.email1 AND v.email2 = NEW.email2 AND v.VinculoActivo = TRUE);
SET g = n+d;
IF(g>0) then
 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =' Solo se admite un vínculo
activo entre 2 personas';
END IF;
END //
DELIMITER ;

-- realizados por Álvaro Caño Soto
DElIMITER //
CREATE OR REPLACE TRIGGER convVincAmistad
BEFORE INSERT ON conversaciones
FOR EACH ROW
BEGIN
DECLARE VinculoActivo BOOLEAN;
SET VinculoActivo  = (SELECT VinculoActivo FROM vinculos_amistades v WHERE
(v.vinculoID = NEW.vinculoID));
if(!VinculoActivo) then
SIGNAL SQLSTATE'45000' SET MESSAGE_TEXT = 'Una conversación solo puede
crearse bajo un vínculo de amistad activo';
END IF;
END //
DELIMITER ;

DElIMITER //
CREATE OR REPLACE TRIGGER intervConver
BEFORE INSERT ON mensajes
FOR EACH ROW
BEGIN
DECLARE fecha_inicio DATE;
DECLARE fecha_fin DATE;
SET fecha_inicio  = (SELECT fecha_inicio FROM conversaciones WHERE
(conversacionID  = new.conversacionID));
SET fecha_fin  = (SELECT fecha_fin FROM conversaciones WHERE
(conversacionID  = new.conversacionID));
if(fecha_inicio > NEW.instante || fecha_fin < NEW.instante) then
SIGNAL SQLSTATE'45000' SET MESSAGE_TEXT = 'El instante en que se genera un
mensaje debe estar dentro del intervalo definido para una conversación';
END IF;
END //
DELIMITER ;

-- otros triggesr realizados por jesus carrascosa
DElIMITER //
create or REPLACE trigger mensajesImutables
BEFORE UPDATE ON mensajes
FOR EACH ROW
BEGIN
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Los mensajes son inmutables';
END //
```

```
DELIMITER ;

DELIMITER //
create or REPLACE trigger mensajesInborrables
BEFORE DELETE ON mensajes
FOR EACH ROW
BEGIN
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Los mensajes no se pueden
borrar';
END //
DELIMITER ;

DELIMITER //
create or REPLACE trigger conversacionesInborrables
BEFORE DELETE ON conversaciones
FOR EACH ROW
BEGIN
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Las conversaciones no se
pueden borrar';
END //
DELIMITER ;

DELIMITER //
create OR REPLACE TRIGGER  mensajesConversacionActiva
BEFORE INSERT ON mensajes
FOR EACH ROW
BEGIN
DECLARE b BOOLEAN;
SET b = (SELECT new_message FROM conversaciones WHERE conversacionID =
NEW.conversacionID);
if(b =FALSE) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'para enviar los mensajes se
necesita una conversacion activa';
END if;
END //
DELIMITER ;

-- trigger para la eliminicaion del usuario
DELIMITER //
create OR REPLACE TRIGGER  eliminarUsuario
BEFORE DELETE ON usuarios
FOR EACH ROW
BEGIN
 DECLARE id VARCHAR(250);
 DECLARE fp INTEGER;
 DECLARE f INTEGER ;
 DECLARE v INTEGER;
 DECLARE ub INTEGER;
 SET id = OLD.email;
 SET fp = (SELECT COUNT(*) FROM fichasDePreferencias  WHERE email =
OLD.email);
 SET f = ( SELECT COUNT(*) FROM fotos  WHERE email = OLD.email);
 SET v = (SELECT COUNT(*) FROM vinculos_amistades   WHERE email1 =
OLD.email OR email2 = OLD.email);
```

```
 set ub = (SELECT COUNT(*) FROM ubicaciones WHERE email = OLD.email);
 if( (fp+f+v+ ub) >0) then
 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'no se puede borrar a este
usuario';
 END if;
 END //

DELIMITER ;

DELIMITER //
create OR REPLACE TRIGGER usuarioActivo1
BEFORE insert ON fotos
FOR EACH ROW
begin
DECLARE b DATE;
DECLARE f DATE;
SET f = (CURDATE());
SET b = (SELECT fechaBaja FROM usuarios WHERE email = NEW.email);
if (b <= f) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'es necesario que el usuario
esté activo';
END if;
END //
DELIMITER ;

DELIMITER //
create OR REPLACE TRIGGER usuarioActivo2
BEFORE delete ON fotos
FOR EACH ROW
begin
DECLARE b DATE;
DECLARE f DATE;
SET f = (CURDATE());
SET b = (SELECT fechaBaja FROM usuarios WHERE email = old.email);
if (b <= f) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'es necesario que el usuario
esté activo';
END if;
END //
DELIMITER ;
DELIMITER //
create OR REPLACE TRIGGER usuarioActivo3
BEFORE update ON fotos
FOR EACH ROW
begin
DECLARE b DATE;
DECLARE f DATE;
SET f = (CURDATE());
SET b = (SELECT fechaBaja FROM usuarios WHERE email = NEW.email);
if (b <= f) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'es necesario que el usuario
esté activo';
END if;
END //
```

```
DELIMITER ;
DELIMITER //
create OR REPLACE TRIGGER usuarioActivo4
BEFORE insert ON vinculos_amistades
FOR EACH ROW
begin
DECLARE b DATE;
DECLARE f DATE;
SET f = (CURDATE());
SET b = (SELECT fechaBaja FROM usuarios WHERE email = NEW.email1 OR email
= NEW.email2);
if (b <= f) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'es necesario que el usuario
esté activo';
END if;
END //
DELIMITER ;
DELIMITER //
create OR REPLACE TRIGGER usuarioActivo5
BEFORE update ON vinculos_amistades
FOR EACH ROW
begin
DECLARE b DATE;
DECLARE f DATE;
SET f = (CURDATE());
SET b = (SELECT fechaBaja FROM usuarios WHERE email = NEW.email1 OR email
= NEW.email2);
if (b <= f) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'es necesario que el usuario
esté activo';
END if;
END //
DELIMITER ;
DELIMITER //
create OR REPLACE TRIGGER usuarioActivo6
BEFORE delete ON vinculos_amistades
FOR EACH ROW
begin
DECLARE b DATE;
DECLARE f DATE;
SET f = (CURDATE());
SET b = (SELECT fechaBaja FROM usuarios WHERE email = old.email1 OR email
= old.email2);
if (b <= f) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'es necesario que el usuario
esté activo';
END if;
END //
DELIMITER ;

DELIMITER //
create OR REPLACE TRIGGER usuarioActivo7
BEFORE update ON fichasDePreferencias
FOR EACH ROW
```

```
begin
DECLARE b DATE;
DECLARE f DATE;
SET f = (CURDATE());
SET b = (SELECT fechaBaja FROM usuarios WHERE email = NEW.email);
if (b <= f) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'es necesario que el usuario
esté activo';
END if;
END //
DELIMITER ;

DELIMITER //
create OR REPLACE TRIGGER usuarioActivo8
BEFORE insert ON fichasDePreferencias
FOR EACH ROW
begin
DECLARE b DATE;
DECLARE f DATE;
SET f = (CURDATE());
SET b = (SELECT fechaBaja FROM usuarios WHERE email = NEW.email);
if (b <= f) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'es necesario que el usuario
esté activo';
END if;
END //
DELIMITER ;
DELIMITER //
create OR REPLACE TRIGGER usuarioActivo9
BEFORE delete ON fichasDePreferencias
FOR EACH ROW
begin
DECLARE b DATE;
DECLARE f DATE;
SET f = (CURDATE());
SET b = (SELECT fechaBaja FROM usuarios WHERE email = old.email);
if (b <= f) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'es necesario que el usuario
esté activo';
END if;
END //
DELIMITER ;
DELIMITER //
create OR REPLACE TRIGGER usuarioActivo10
BEFORE update ON aficiones_usuarios
FOR EACH ROW
begin
DECLARE b DATE;
DECLARE f DATE;
SET f = (CURDATE());
SET b = (SELECT fechaBaja FROM usuarios WHERE email = NEW.email);
if (b <= f) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'es necesario que el usuario
esté activo';
```

```
END if;
END //
DELIMITER ;
DELIMITER //
create OR REPLACE TRIGGER usuarioActivo10
BEFORE insert ON aficiones_usuarios
FOR EACH ROW
begin
DECLARE b DATE;
DECLARE f DATE;
SET f = (CURDATE());
SET b = (SELECT fechaBaja FROM usuarios WHERE email = NEW.email);
if (b <= f) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'es necesario que el usuario
esté activo';
END if;
END //
DELIMITER ;
DELIMITER //
create OR REPLACE TRIGGER usuarioActivo11
BEFORE update ON aficiones_usuarios
FOR EACH ROW
begin
DECLARE b DATE;
DECLARE f DATE;
SET f = (CURDATE());
SET b = (SELECT fechaBaja FROM usuarios WHERE email = NEW.email);
if (b <= f) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'es necesario que el usuario
esté activo';
END if;
END //
DELIMITER ;
DELIMITER //
create OR REPLACE TRIGGER usuarioActivo12
BEFORE delete ON aficiones_usuarios
FOR EACH ROW
begin
DECLARE b DATE;
DECLARE f DATE;
SET f = (CURDATE());
SET b = (SELECT fechaBaja FROM usuarios WHERE email = old.email);
if (b <= f) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'es necesario que el usuario
esté activo';
END if;
END //
DELIMITER ;

DELIMITER //
create OR REPLACE TRIGGER aficcionEnUso
BEFORE DELETE ON aficiones
FOR EACH ROW
BEGIN
```

```
DECLARE u INTEGER;
DECLARE p INTEGER;
DECLARE s INTEGER;

SET u = (SELECT COUNT(*) FROM (aficiones NATURAL JOIN aficiones_usuarios)
WHERE aficionID = OLD.aficionID);
SET p = (SELECT COUNT(*) FROM (aficiones NATURAL JOIN
aficiones_preferidas)  WHERE aficionID = OLD.aficionID);

SET s = u +p;
if(s >0) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'a aficion está en uso y no se
puede borrar';
END if;
END //
DELIMITER ;

DELIMITER //
create OR REPLACE TRIGGER minimoUbicaciones
BEFORE DELETE ON ubicaciones
FOR EACH ROW
BEGIN
DECLARE ut INTEGER;
SET ut = (SELECT COUNT(*) FROM ubicaciones WHERE email = OLD.email);
if(ut<=1) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'un usuario tiene que tener al
menos una aficion';

END if ;
END //
DELIMITER ;

DELIMITER //
create OR REPLACE TRIGGER FichaConAficiones
BEFORE DELETE ON fichasDePreferencias
FOR EACH ROW
BEGIN
DECLARE n INTEGER;
SET n = (SELECT COUNT(*) FROM aficiones_preferidas WHERE fichaID =
OLD.fichaID);
if(n >0) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'No se puede borrar una ficha
con aficiones vinculadas';
END if ;
END //
DELIMITER ;
 -- coherencia fecha de baja y estaActivo

DELIMITER //
create OR REPLACE TRIGGER coherenciaFechaBajaEstaActivo1
BEFORE UPDATE ON usuarios
FOR EACH ROW
BEGIN
```

```
if(New.fechaBaja  is NULL AND New.estaActivo = FALSE)then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'la fecha de baja y el
estaActivo se contradicen';
END if ;
IF(New.fechaBaja is not NULL AND New.estaActivo = TRUE) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'la fecha de baja y el
estaActivo se contradicen';
END if ;
END //
DELIMITER ;
DELIMITER //
create OR REPLACE TRIGGER coherenciaFechaBajaEstaActivo2
BEFORE INSERT ON usuarios
FOR EACH ROW
BEGIN

if(New.fechaBaja  is NULL AND New.estaActivo = FALSE)then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'la fecha de baja y el
estaActivo se contradicen';
END if ;
IF(New.fechaBaja is not NULL AND New.estaActivo = TRUE) then
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'la fecha de baja y el
estaActivo se contradicen';
END if ;
END //
DELIMITER ;
```