Proyecto de diseño de una Base de Datos

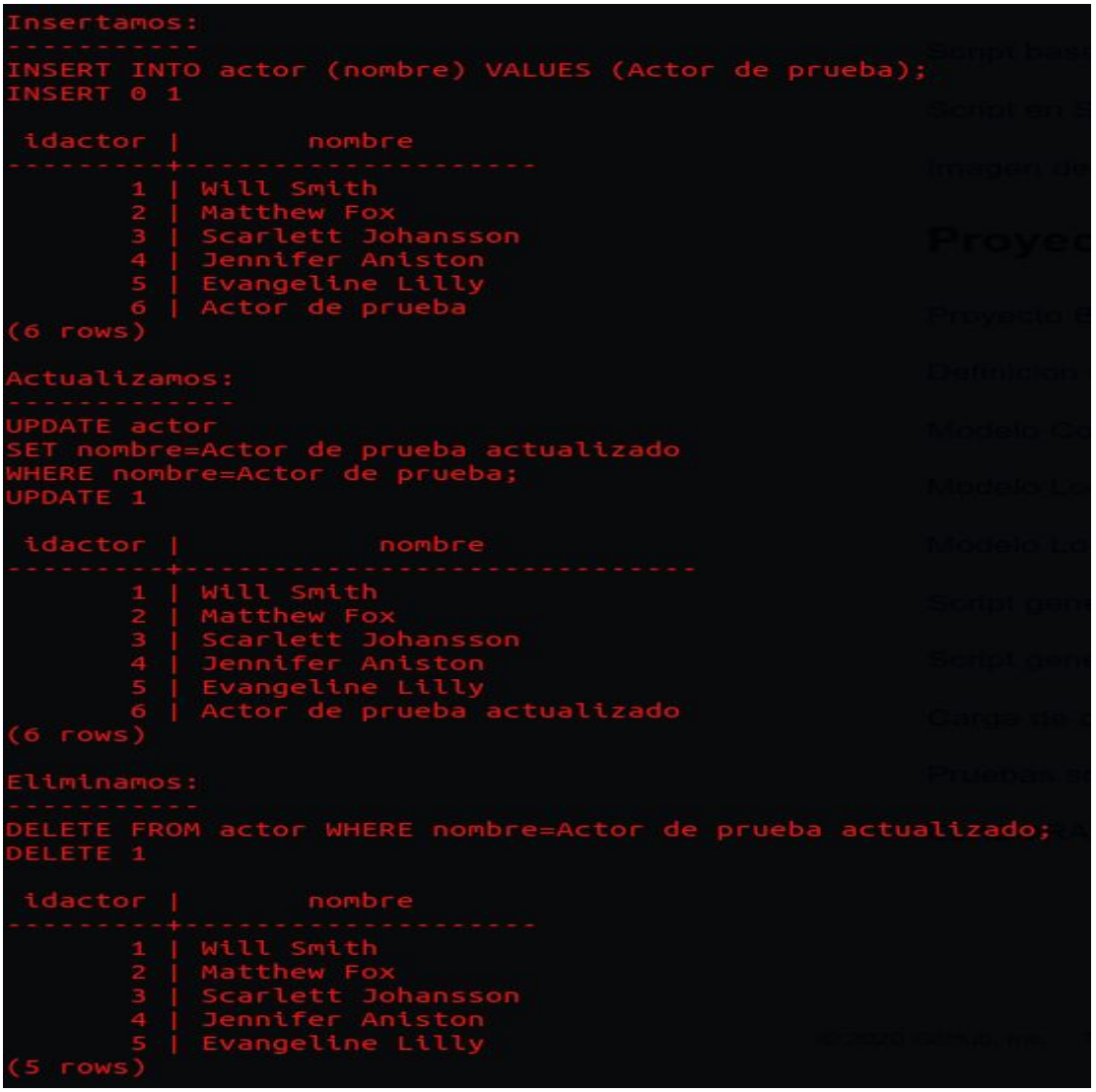
Servicio de video streaming OAN
Pruebas inserción/actualización/borrado/consultas + funcionamiento triggers/checks

Vlad Comanescu
Ruymán Rodríguez Martín
Diego Machín Guardia

Pruebas inserción,actualización,borrado y consultas

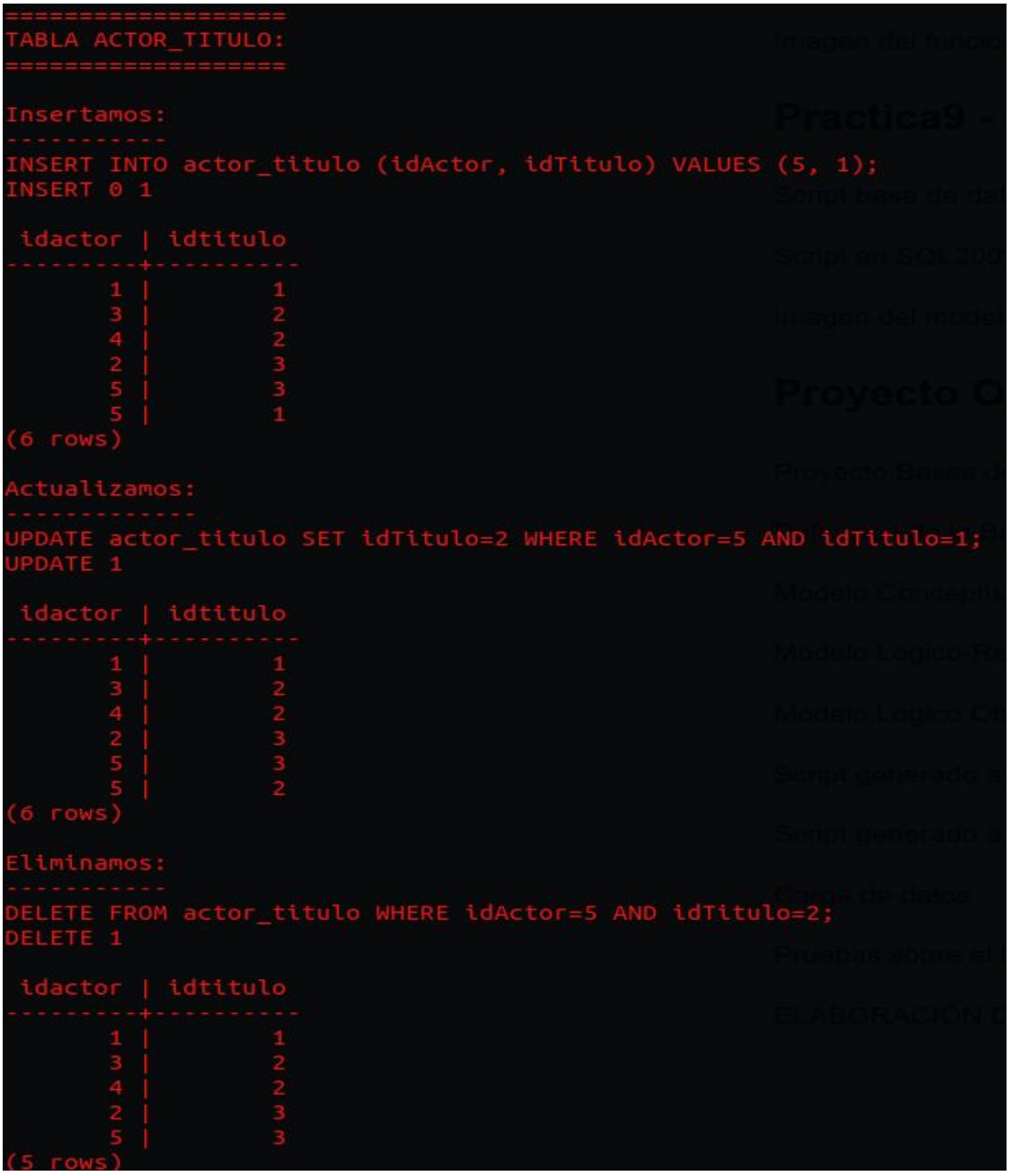
En esta sección haremos pruebas de las diferentes operaciones de inserción,actualización,borrado y consultas sobre las diferentes tablas de nuestra base de datos.Asimismo,también comprobaremos la integridad referencial que se da entre las diferentes tablas y cómo afectaría los cambios en una tabla a otra/s tablas.

Tenemos la tabla **Actor** y **actor_titulo**.La primera de ellas guarda la información de los distintos actores(id y nombre) y la segunda relaciona a estos actores con los **títulos**.



```
Insertamos:
-----------
INSERT INTO actor (nombre) VALUES (Actor de prueba);
INSERT 0 1

 idactor |        nombre
---------+----------------------
       1 | Will Smith
       2 | Matthew Fox
       3 | Scarlett Johansson
       4 | Jennifer Aniston
       5 | Evangeline Lilly
       6 | Actor de prueba
(6 rows)

Actualizamos:
-------------
UPDATE actor
SET nombre=Actor de prueba actualizado
WHERE nombre=Actor de prueba;
UPDATE 1

 idactor |            nombre
---------+------------------------------
       1 | Will Smith
       2 | Matthew Fox
       3 | Scarlett Johansson
       4 | Jennifer Aniston
       5 | Evangeline Lilly
       6 | Actor de prueba actualizado
(6 rows)

Eliminamos:
-----------
DELETE FROM actor WHERE nombre=Actor de prueba actualizado;
DELETE 1

 idactor |        nombre
---------+----------------------
       1 | Will Smith
       2 | Matthew Fox
       3 | Scarlett Johansson
       4 | Jennifer Aniston
       5 | Evangeline Lilly
(5 rows)
```

Realizamos las diferentes operaciones de inserción,actualización y borrado sobre la tabla **actor.**



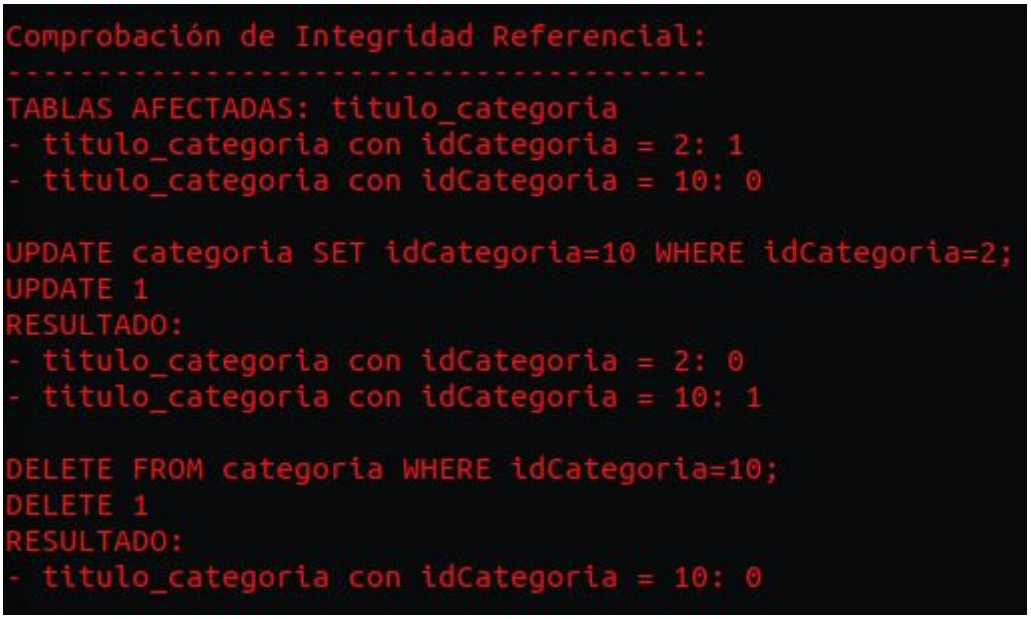
Realizamos las diferentes operaciones de inserción,actualización y borrado sobre la tabla **actor_titulo** modificando el **idActor** que es clave ajena de la tabla actor e **idTitulo** que es clave ajena de la tabla **titulo.**
Una actualización o borrado en la tabla **actor** propagara una actualización o borrado en cascada en la tabla **actor_titulo.**

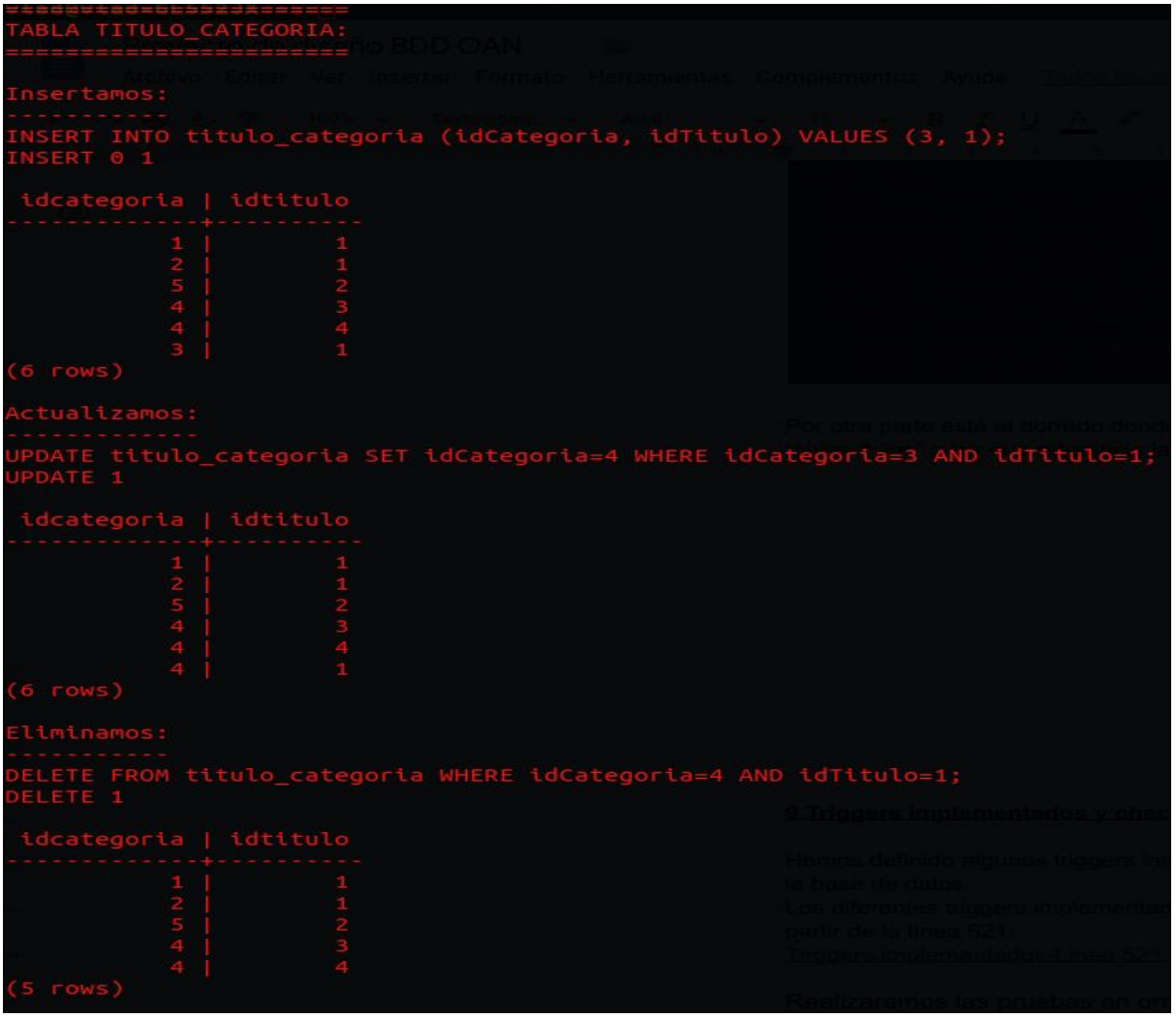
```
Comprobación de Integridad Referencial:
----------------------------------------
TABLAS AFECTADAS: actor_titulo
- actor_titulo con idActor = 2: 1
- actor_titulo con idActor = 10: 0

UPDATE actor SET idActor=10 WHERE idActor=2;
UPDATE 1
RESULTADO:
- actor_titulo con idActor = 2: 0
- actor_titulo con idActor = 10: 1

DELETE FROM actor WHERE idActor=10;
DELETE 1
RESULTADO:
- actor_titulo con idActor = 10: 0
```

Vemos que también se mantiene la integridad referencial actualizando la clave primaria en la tabla **actor** y modifica también en la tabla que se referencia **actor_titulo**.Si se elimina una clave primaria en la tabla **actor** también se modifica la clave primaria a la que referencia en la tabla **actor_titulo.**

```
================
TABLA CATEGORIA:
================

Insertamos:
-----------
INSERT INTO categoria (nombre, descripcion) VALUES (cat1, categoria de prueba insercion);
INSERT 0 1

 idcategoria |     nombre      |          descripcion
-------------+-----------------+-------------------------------
           1 | Accion          | Muchas explosiones y disparos
           2 | Comedia         | Jajajaja
           3 | Artes Marciales | Pim Pam Pum
           4 | Drama           | Chiquita angustia
           5 | Doc. Naturaleza | Vida salvaje
           6 | cat1            | categoria de prueba insercion
(6 rows)

Actualizamos:
-------------
UPDATE categoria
SET nombre=cat1update, descripcion=categoria de prueba actualizacion
WHERE nombre=cat1;
UPDATE 1

 idcategoria |     nombre      |            descripcion
-------------+-----------------+---------------------------------
           1 | Accion          | Muchas explosiones y disparos
           2 | Comedia         | Jajajaja
           3 | Artes Marciales | Pim Pam Pum
           4 | Drama           | Chiquita angustia
           5 | Doc. Naturaleza | Vida salvaje
           6 | cat1update      | categoria de prueba actualizacion
(6 rows)

Eliminamos:
-----------
DELETE FROM categoria WHERE nombre=cat1update;
DELETE 1

 idcategoria |     nombre      |          descripcion
-------------+-----------------+-------------------------------
           1 | Accion          | Muchas explosiones y disparos
           2 | Comedia         | Jajajaja
           3 | Artes Marciales | Pim Pam Pum
           4 | Drama           | Chiquita angustia
           5 | Doc. Naturaleza | Vida salvaje
(5 rows)
```

En la tabla **Categoria** probamos insertando,actualizando y borrando datos de la tabla.
En esta tabla se guardarán datos de la categoría de un título,id,nombre y una descripción de la misma.
Luego en la tabla **titulo_categoria** referenciada que contendrá como clave primaria el idCategoria ocurrirá lo mismo que en el ejemplo anterior manteniéndose la integridad referencial.
Una actualización o borrado en la tabla **Categoria** propagara una actualización o borrado en cascada en la tabla **titulo_categoria.**

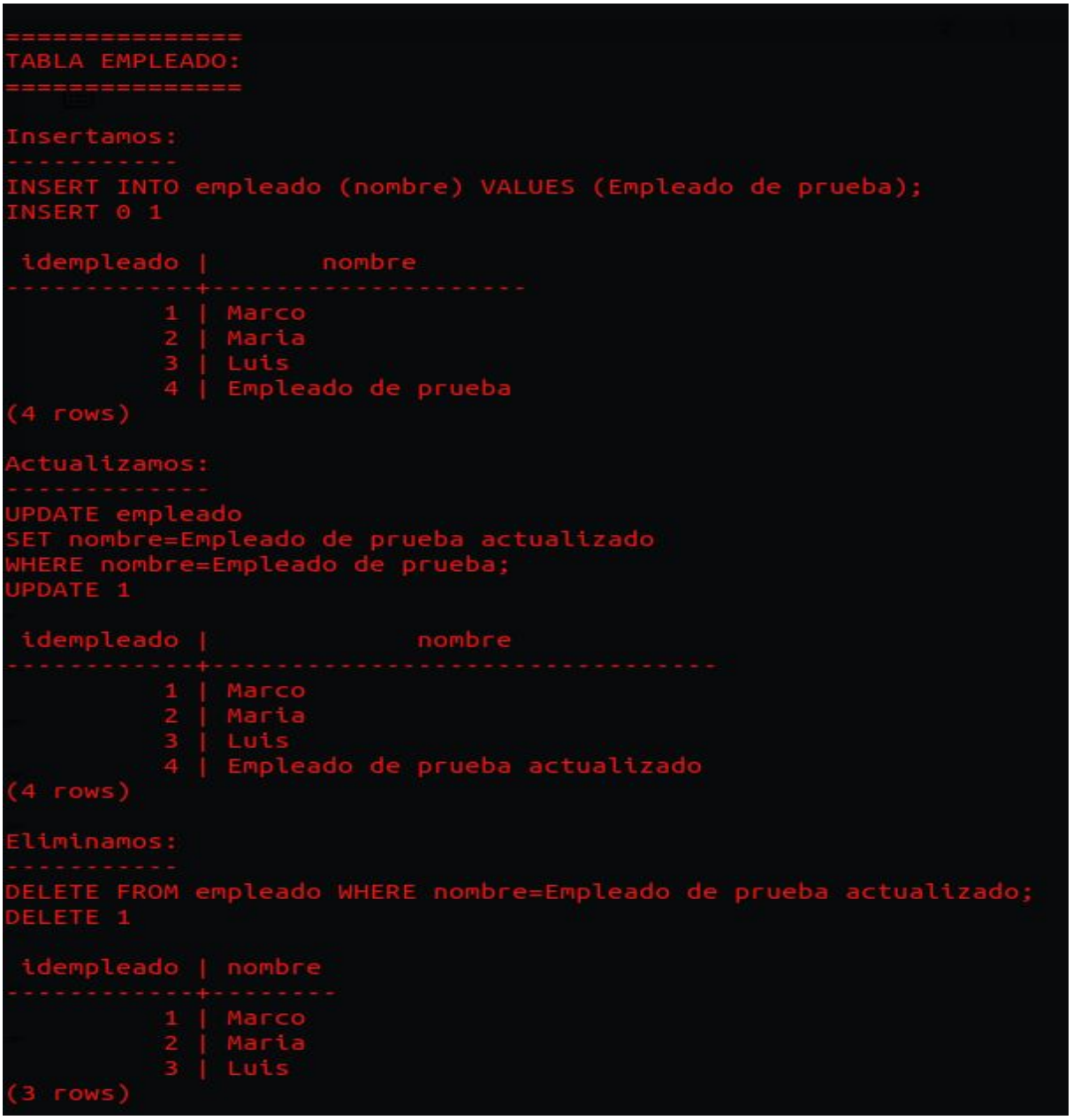




La tabla **titulo_categoria** contendra la informacion necesaria para relacionar las categorias existentes con los titulos.Es decir un titulo identificado por un **idtitulo** tendra una o varias categorías de las que formaria parte.

```
================
TABLA DESCARGAS:
================

Insertamos:
-----------
INSERT INTO descargas (idTitulo, fecha_caducidad, idPerfil) VALUES (1, CURRENT_DATE + 1, 4);
INSERT 0 1

 idtitulo | fecha_caducidad | idperfil
----------+-----------------+----------
        3 | 2020-01-27      |        1
        3 | 2020-01-30      |        2
        2 | 2020-02-05      |        4
        1 | 2020-02-01      |        4
(4 rows)

Actualizamos:
-------------
UPDATE descargas
SET fecha_caducidad=CURRENT_DATE + 10
WHERE idPerfil=4 AND idTitulo=1;
UPDATE 1

 idtitulo | fecha_caducidad | idperfil
----------+-----------------+----------
        3 | 2020-01-27      |        1
        3 | 2020-01-30      |        2
        2 | 2020-02-05      |        4
        1 | 2020-02-10      |        4
(4 rows)

Eliminamos:
-----------
DELETE FROM descargas WHERE idPerfil=4 AND idTitulo=1;
DELETE 1

 idtitulo | fecha_caducidad | idperfil
----------+-----------------+----------
        3 | 2020-01-27      |        1
        3 | 2020-01-30      |        2
        2 | 2020-02-05      |        4
(3 rows)
```

La tabla **descargas** contendrá la/las descarga/s de los distintos perfiles de usuarios es decir un perfil tendrá uno o múltiples descargas cada una con su fecha de caducidad.
Probamos insertando,actualizando modificando la fecha de caducidad de un título de un perfil y borrar.



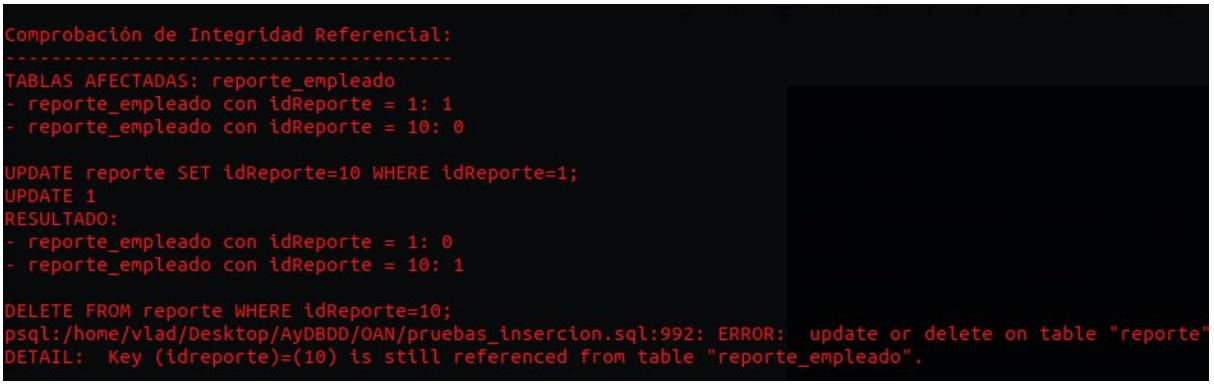
La tabla **empleado** contendra informacion sobre los empleados de nuestra empresa OAN,habrá un identificador único y el nombre de cada empleado.
Una actualización en la tabla **empleado** propagara una actualización en cascada en la tabla **reporte_empleado.**Sin embargo un borrado en la tabla **empleado** está restriccionado y por lo tanto daría error en caso de que se produjera.

```
==============
TABLA REPORTE:
==============

Insertamos:
-----------
INSERT INTO reporte (descripcion, estado, email)
VALUES (Prueba, En Pruebas, victoria.md@gmail.com);
INSERT 0 1

 idreporte |          descripcion           |   estado   |              email
-----------+--------------------------------+------------+---------------------------------
         1 | Error al reproducir el video   | Cerrado    | antonio.gutierrez1984@gmail.com
         2 | A veces no se escucha el audio | Abierto    | julia.almeida@gmail.com
         3 | No me deja descargar           | Abierto    | julia.almeida@gmail.com
         4 | Prueba                         | En Pruebas | victoria.md@gmail.com
(4 rows)

Actualizamos:
-------------
UPDATE reporte
SET estado=Actualizado
WHERE email=victoria.md@gmail.com AND descripcion=Prueba;
UPDATE 1

 idreporte |          descripcion           |   estado    |              email
-----------+--------------------------------+-------------+---------------------------------
         1 | Error al reproducir el video   | Cerrado     | antonio.gutierrez1984@gmail.com
         2 | A veces no se escucha el audio | Abierto     | julia.almeida@gmail.com
         3 | No me deja descargar           | Abierto     | julia.almeida@gmail.com
         4 | Prueba                         | Actualizado | victoria.md@gmail.com
(4 rows)

Eliminamos:
-----------
DELETE FROM reporte WHERE email=victoria.md@gmail.com AND descripcion=Prueba;
DELETE 1

 idreporte |          descripcion           | estado  |              email
-----------+--------------------------------+---------+---------------------------------
         1 | Error al reproducir el video   | Cerrado | antonio.gutierrez1984@gmail.com
         2 | A veces no se escucha el audio | Abierto | julia.almeida@gmail.com
         3 | No me deja descargar           | Abierto | julia.almeida@gmail.com
(3 rows)
```

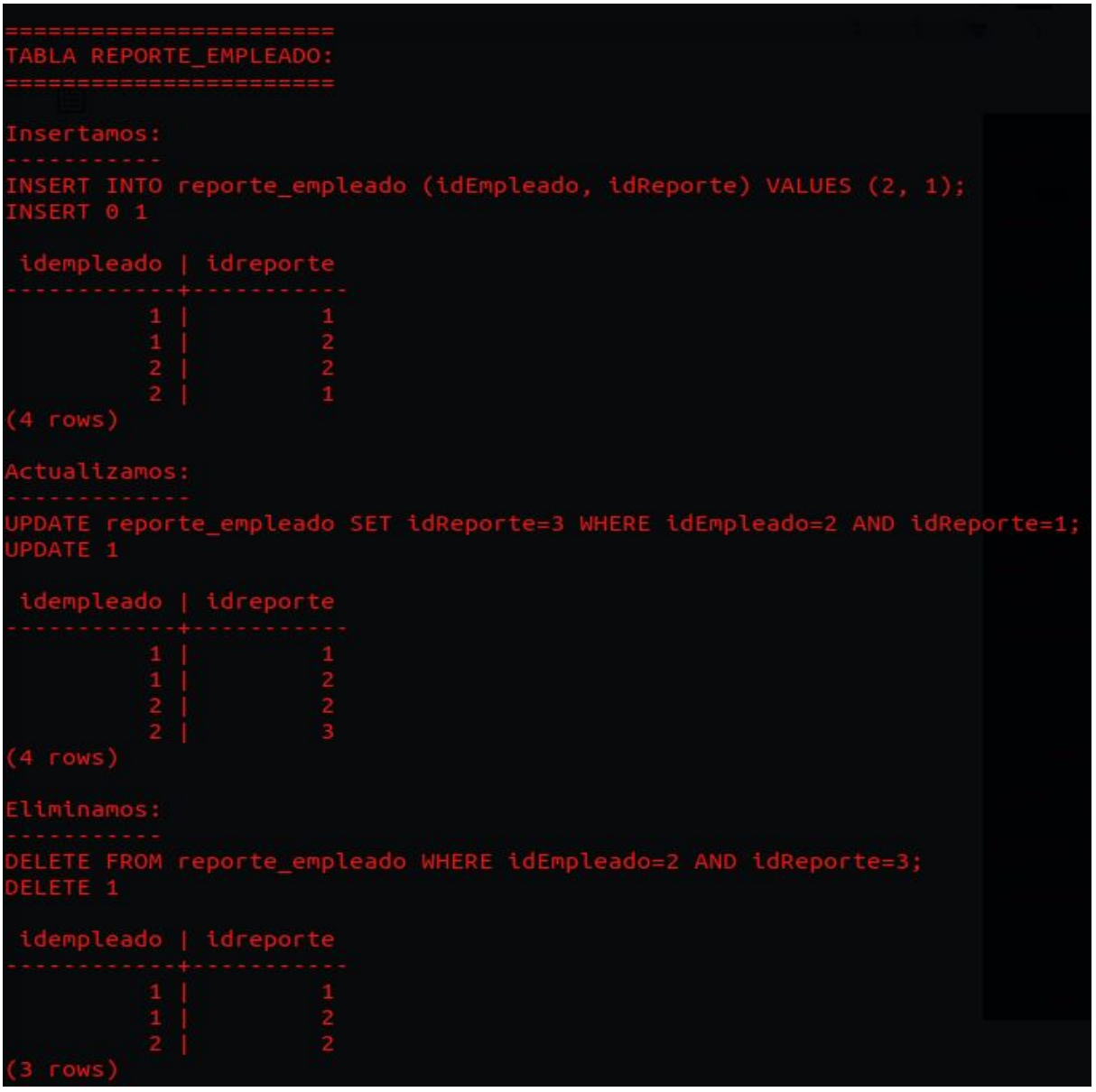
La tabla **reporte** contendrá información sobre los diferentes reportes que puede realizar un usuario sobre alguna incidencia que le haya ocurrido.Tendrá un id único,una descripción del reporte,el estado de dicho reporte y el correo del usuario que haya empezado el reporte.

Una actualización en la tabla **reporte** propagara una actualización en cascada en la tabla **reporte_empleado.**Sin embargo un borrado en la tabla **reporte** está restriccionado y por lo tanto daría error en caso de que se produjera.

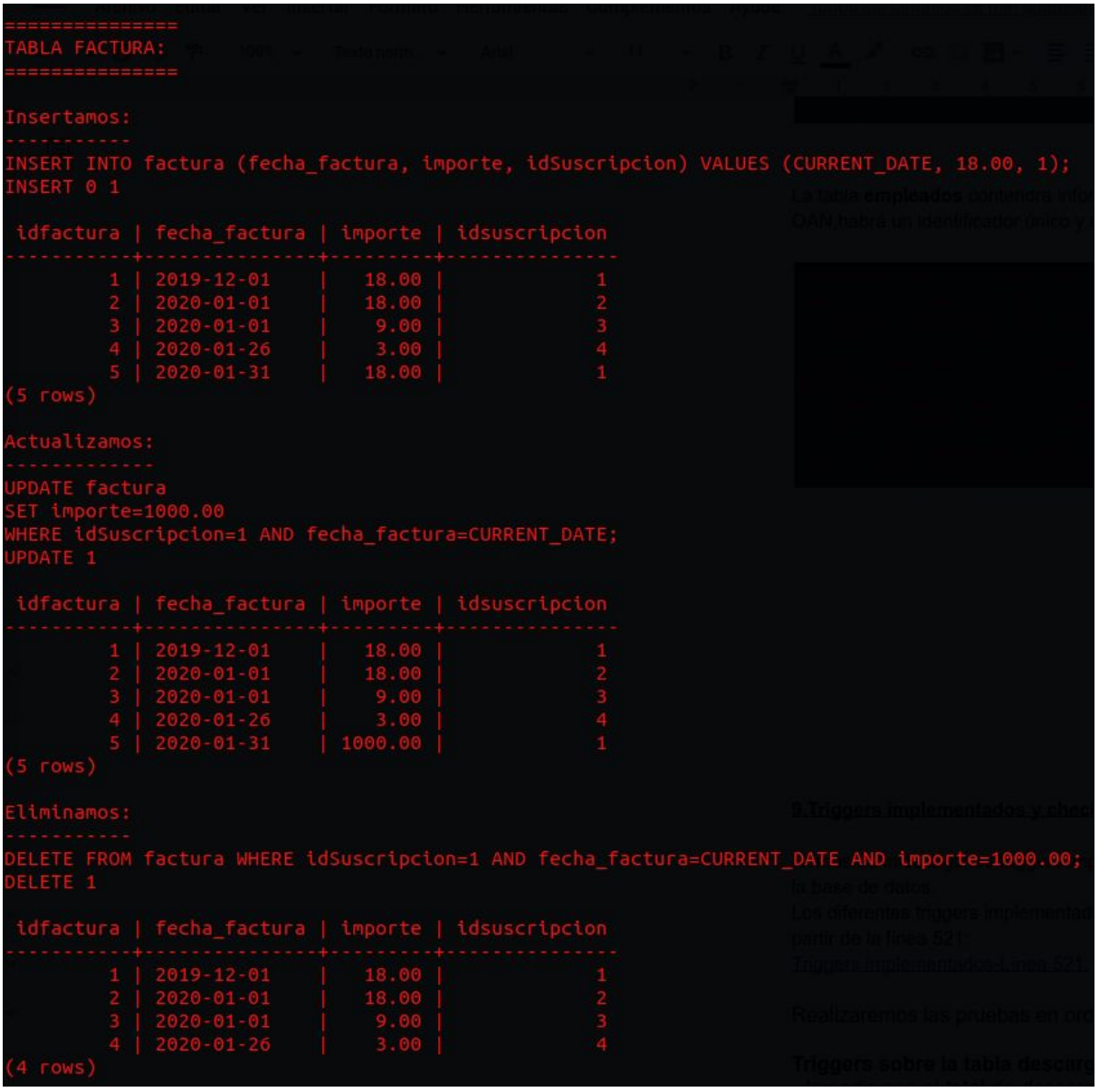
```
Comprobación de Integridad Referencial:
---------------------------------------
TABLAS AFECTADAS: reporte_empleado
- reporte_empleado con idEmpleado = 1: 2
- reporte_empleado con idEmpleado = 10: 0

UPDATE empleado SET idEmpleado=10 WHERE idEmpleado=1;
UPDATE 1
RESULTADO:
- reporte_empleado con idEmpleado = 1: 0
- reporte_empleado con idEmpleado = 10: 2

DELETE FROM empleado WHERE idEmpleado=10;
psql:/home/vlad/Desktop/AyDBDD/OAN/pruebas_insercion.sql:298: ERROR:
DETAIL:  Key (idempleado)=(10) is still referenced from table "report
```



Comprobamos que las actualizaciones sobre la tabla **reporte** o **empleado** no daría problemas en la tabla **reporte_empleado**, sin embargo, un borrado si que daría error ya que esta restriccionado.



La tabla **reporte_empleado** contendrá la asignación de un reporte hecho por un usuario a un empleado de la empresa.Se puede asignar a un unico empleado multiples reportes de distintos usuarios.



```
===============
TABLA FACTURA:
===============

Insertamos:
-----------
INSERT INTO factura (fecha_factura, importe, idSuscripcion) VALUES (CURRENT_DATE, 18.00, 1);
INSERT 0 1

 idfactura | fecha_factura | importe | idsuscripcion
-----------+---------------+---------+---------------
         1 | 2019-12-01    |   18.00 |             1
         2 | 2020-01-01    |   18.00 |             2
         3 | 2020-01-01    |    9.00 |             3
         4 | 2020-01-26    |    3.00 |             4
         5 | 2020-01-31    |   18.00 |             1
(5 rows)

Actualizamos:
-------------
UPDATE factura
SET importe=1000.00
WHERE idSuscripcion=1 AND fecha_factura=CURRENT_DATE;
UPDATE 1

 idfactura | fecha_factura | importe | idsuscripcion
-----------+---------------+---------+---------------
         1 | 2019-12-01    |   18.00 |             1
         2 | 2020-01-01    |   18.00 |             2
         3 | 2020-01-01    |    9.00 |             3
         4 | 2020-01-26    |    3.00 |             4
         5 | 2020-01-31    | 1000.00 |             1
(5 rows)

Eliminamos:
-----------
DELETE FROM factura WHERE idSuscripcion=1 AND fecha_factura=CURRENT_DATE AND importe=1000.00;
DELETE 1

 idfactura | fecha_factura | importe | idsuscripcion
-----------+---------------+---------+---------------
         1 | 2019-12-01    |   18.00 |             1
         2 | 2020-01-01    |   18.00 |             2
         3 | 2020-01-01    |    9.00 |             3
         4 | 2020-01-26    |    3.00 |             4
(4 rows)
```

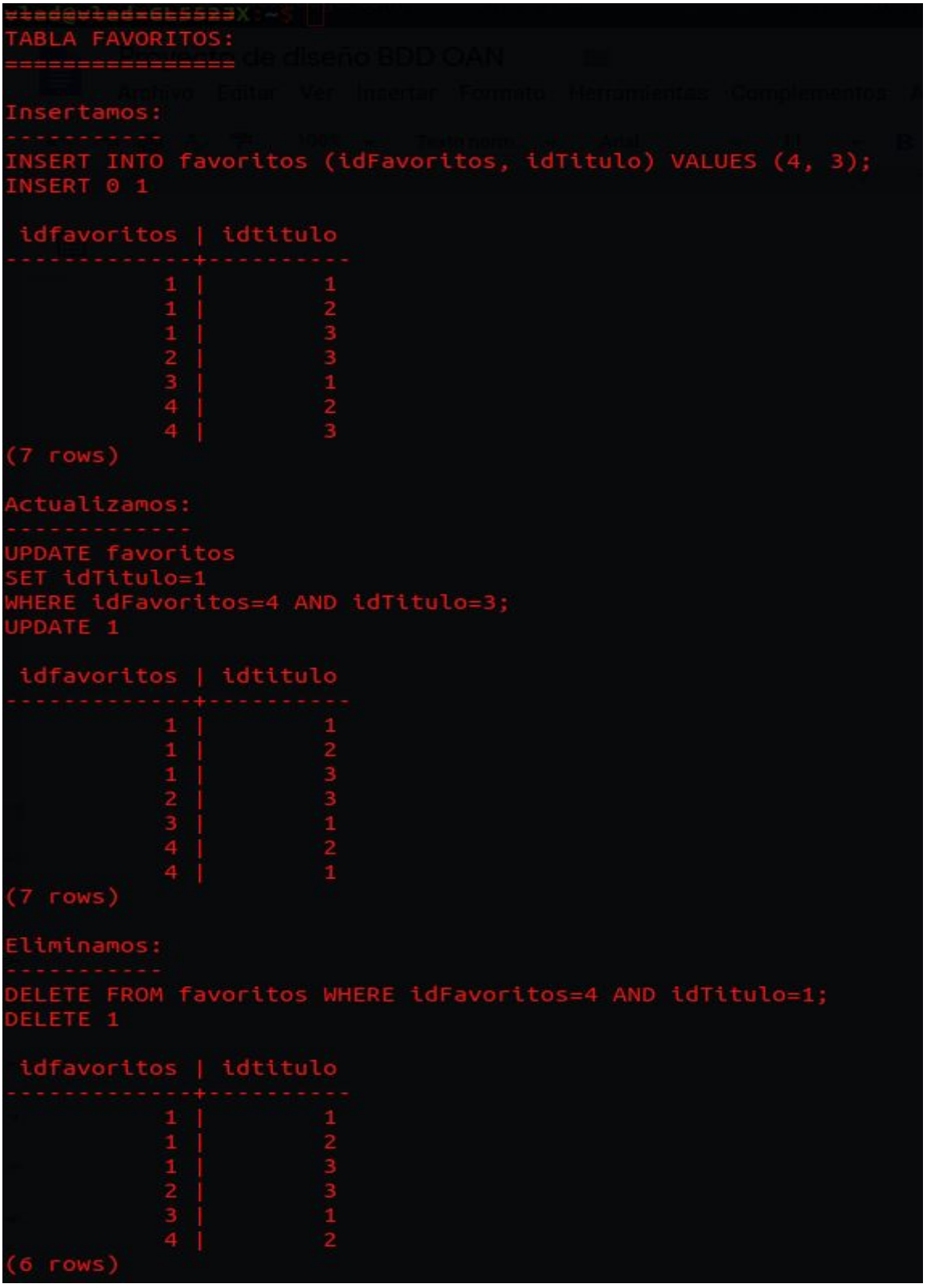
La tabla **factura** contendrá información sobre las distintas facturas de las distintas suscripciones, cada una con su fecha de facturación e importe,referenciando al **idsuscripcion** sobre el cual se ha realizado la factura.

```
============
TABLA PAGO:
============

Insertamos:
-----------
INSERT INTO pago (fecha, metodo_pago, num_tarjeta, CVC, idFactura)
VALUES (CURRENT_DATE, Tajerta, tarjeta de prueba, 111, 3);
INSERT 0 1

 idpago |   fecha    | metodo_pago | nombre |     num_tarjeta     | cvc | fecha_vencimiento | tipo_pago | idfactura
--------+------------+-------------+--------+---------------------+-----+-------------------+-----------+-----------
      1 | 2019-12-02 | Tajerta     |        | 1111 2222 3333 4444 | 123 |                   |           |         1
      2 | 2020-01-01 | Tajerta     |        | 1111 2222 3333 4444 | 123 |                   |           |         2
      3 | 2020-01-26 | Tajerta     |        | 4000 2000 5000 1234 | 432 |                   |           |         4
      4 | 2020-01-31 | Tajerta     |        | tarjeta de prueba   | 111 |                   |           |         3
(4 rows)

Actualizamos:
-------------
UPDATE pago
SET fecha=CURRENT_DATE-5
WHERE idFactura=3;
UPDATE 1

 idpago |   fecha    | metodo_pago | nombre |     num_tarjeta     | cvc | fecha_vencimiento | tipo_pago | idfactura
--------+------------+-------------+--------+---------------------+-----+-------------------+-----------+-----------
      1 | 2019-12-02 | Tajerta     |        | 1111 2222 3333 4444 | 123 |                   |           |         1
      2 | 2020-01-01 | Tajerta     |        | 1111 2222 3333 4444 | 123 |                   |           |         2
      3 | 2020-01-26 | Tajerta     |        | 4000 2000 5000 1234 | 432 |                   |           |         4
      4 | 2020-01-26 | Tajerta     |        | tarjeta de prueba   | 111 |                   |           |         3
(4 rows)

Eliminamos:
-----------
DELETE FROM pago WHERE idFactura=3;
DELETE 1

 idpago |   fecha    | metodo_pago | nombre |     num_tarjeta     | cvc | fecha_vencimiento | tipo_pago | idfactura
--------+------------+-------------+--------+---------------------+-----+-------------------+-----------+-----------
      1 | 2019-12-02 | Tajerta     |        | 1111 2222 3333 4444 | 123 |                   |           |         1
      2 | 2020-01-01 | Tajerta     |        | 1111 2222 3333 4444 | 123 |                   |           |         2
      3 | 2020-01-26 | Tajerta     |        | 4000 2000 5000 1234 | 432 |                   |           |         4
(3 rows)
```

La tabla **pago** contendrá información sobre los distintos pagos realizados en referencia a las distintas facturas de las distintas suscripciones.La informacion que contiene sera fecha de pago,método de pago,nombre, número de tarjeta, cvc, fecha vencimiento, tipo de pago. Procedemos a insertar,actualizar y borrar algunas datos de esta tabla como se puede ver en la captura de encima.

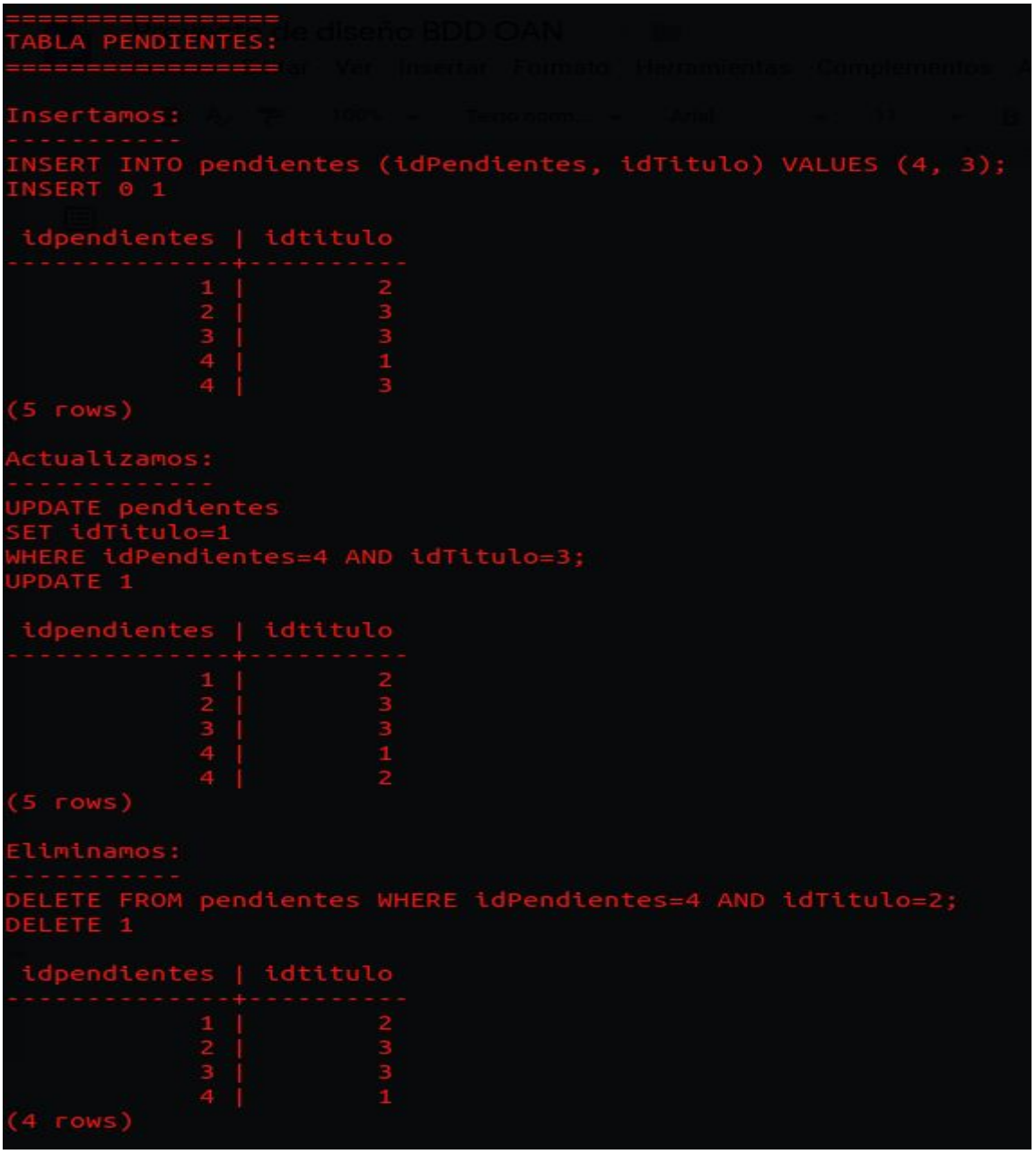
```
Comprobación de Integridad Referencial:
---------------------------------------
TABLAS AFECTADAS: pago
- pago con idFactura = 1: 1
- pago con idFactura = 10: 0

UPDATE factura SET idFactura=10 WHERE idFactura=1;
UPDATE 1
RESULTADO:
- pago con idFactura = 1: 0
- pago con idFactura = 10: 1

DELETE FROM factura WHERE idFactura=10;
psql:/home/vlad/Desktop/AyDBDD/OAN/pruebas_insercion.sql:368: ERROR:  update or delete on table
DETAIL:  Key (idfactura)=(10) is still referenced from table "pago".
```

La tabla **factura** tiene puesta las restricciones de actualización y borrado en cascada.Una actualización o borrado en la tabla **factura** propagara una actualización o borrado en cascada en la tabla **pago.**

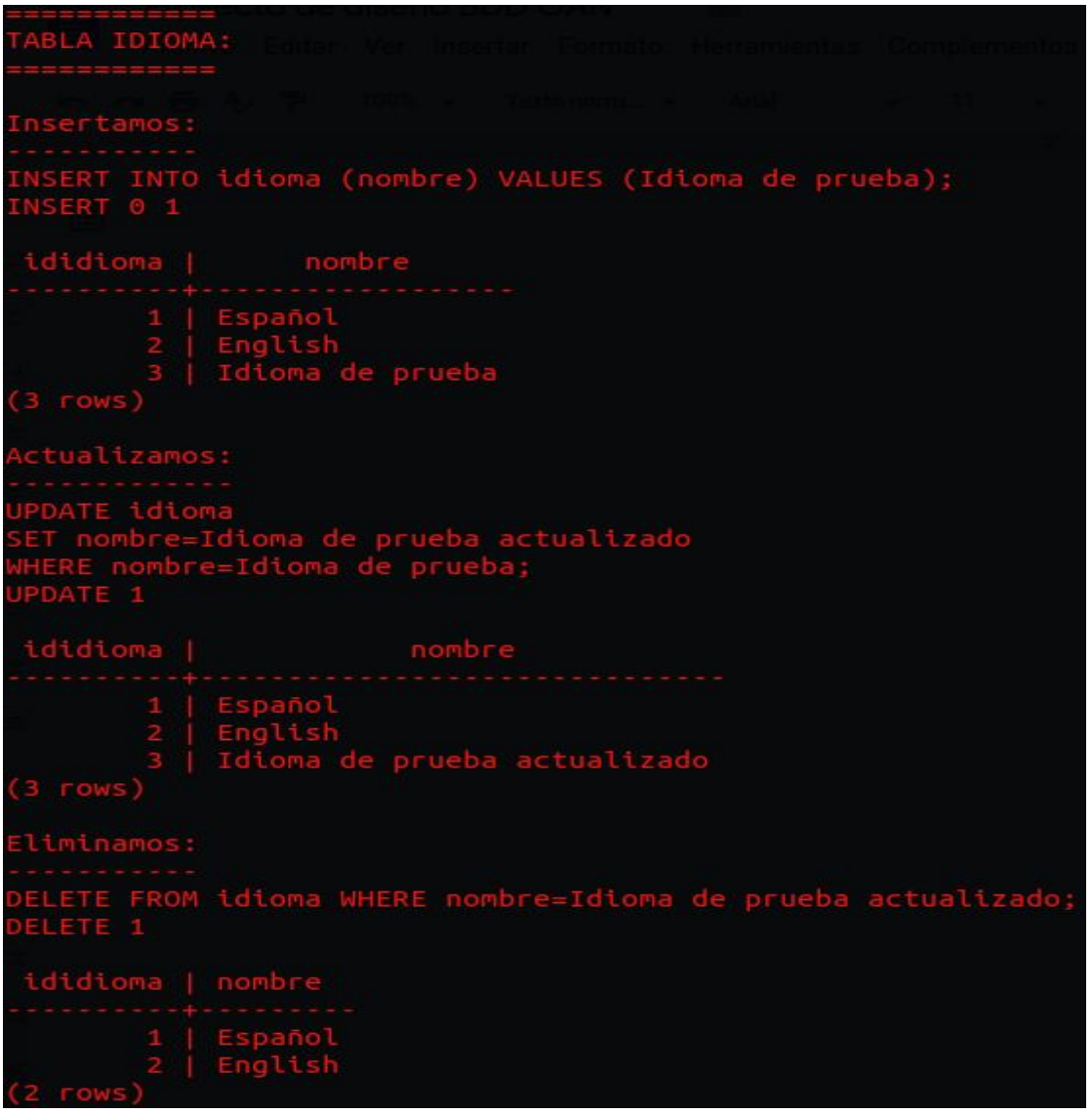


```
TABLA FAVORITOS:
================

Insertamos:
-----------
INSERT INTO favoritos (idFavoritos, idTitulo) VALUES (4, 3);
INSERT 0 1

 idfavoritos | idtitulo
-------------+----------
           1 |        1
           1 |        2
           1 |        3
           2 |        3
           3 |        1
           4 |        2
           4 |        3
(7 rows)

Actualizamos:
-------------
UPDATE favoritos
SET idTitulo=1
WHERE idFavoritos=4 AND idTitulo=3;
UPDATE 1

 idfavoritos | idtitulo
-------------+----------
           1 |        1
           1 |        2
           1 |        3
           2 |        3
           3 |        1
           4 |        2
           4 |        1
(7 rows)

Eliminamos:
-----------
DELETE FROM favoritos WHERE idFavoritos=4 AND idTitulo=1;
DELETE 1

 idfavoritos | idtitulo
-------------+----------
           1 |        1
           1 |        2
           1 |        3
           2 |        3
           3 |        1
           4 |        2
(6 rows)
```

La tabla **favoritos** contendra informacion relacionada con aquellos titulos que un perfil de un usuario tendra como favoritos.Es decir el perfil1 tendrá como favorito(idFavoritos) el 4 que tiene puesta como títulos favoritos el título 1,2.
Podemos eliminar o actualizar un título de la lista de favoritos.



En la tabla **pendientes** vamos a encontrar información relacionada con aquellos títulos que un perfil de un usuario tendrá como pendientes de ver.Es decir el perfil1 tendrá como pendiente(idpendientes) el 4 que tiene puesta como títulos pendientes el título 1 y el 3.
Podemos eliminar o actualizar un título de la lista de pendientes.



La tabla **idioma** contendra informacion relevante a los distintos idiomas que tendrá un título.Tendrán una identificación única y un nombre y nos permitirán definir con más facilidad los idiomas de cada título.

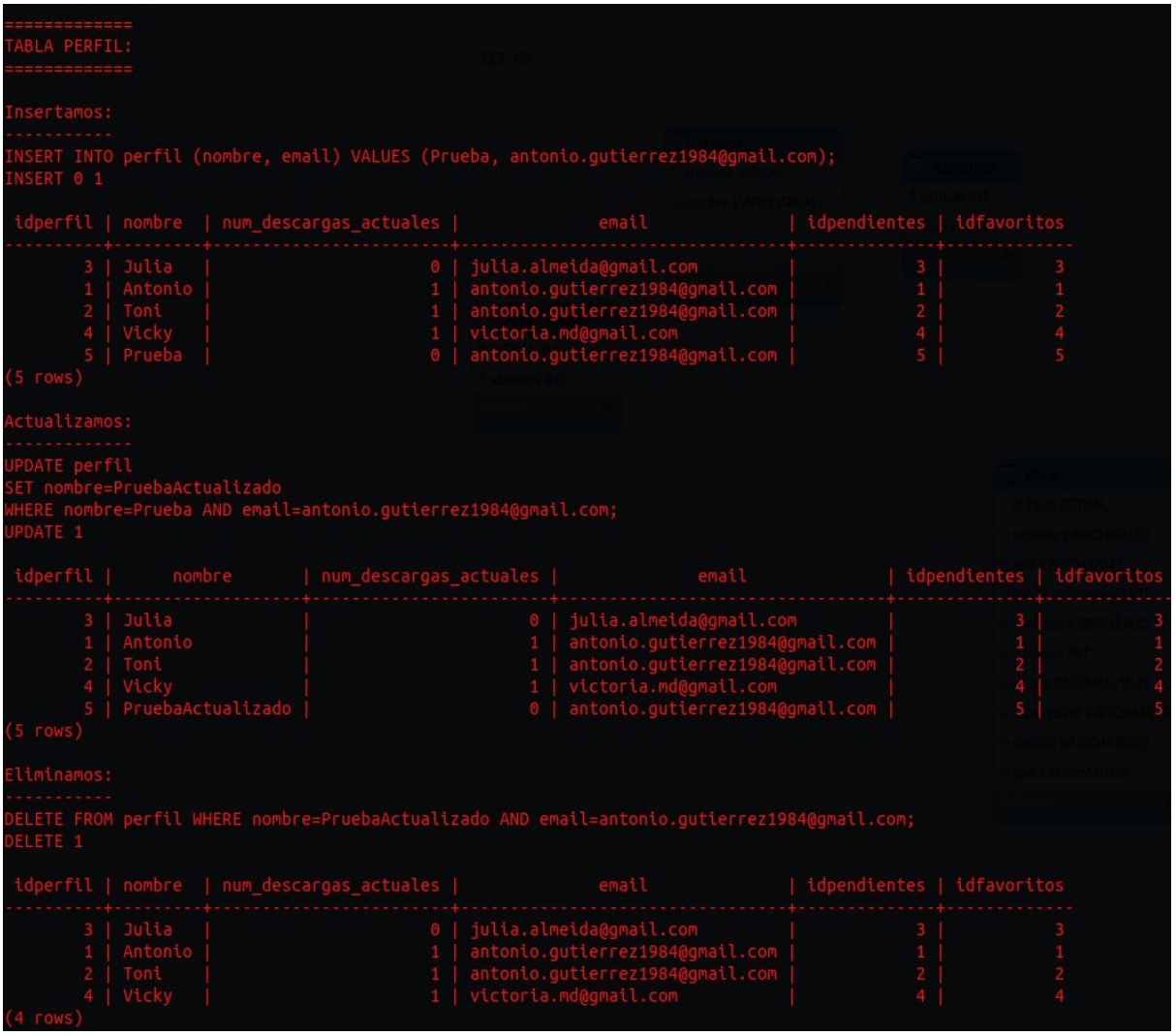
Una actualización o borrado en la tabla **idioma** propagara una actualización en cascada en la tabla **subtitulo** y **titulo_idioma.**

```
Comprobación de Integridad Referencial:
----------------------------------------
TABLAS AFECTADAS: titulo_idioma, subtitulo
- titulo_idioma con idIdioma = 2: 2
- titulo_idioma con idIdioma = 10: 0
- subtitulo con idIdioma = 2: 2
- subtitulo con idIdioma = 10: 0

UPDATE idioma SET idIdioma=10 WHERE idIdioma=2;
UPDATE 1
RESULTADO:
- titulo_idioma con idIdioma = 2: 0
- titulo_idioma con idIdioma = 10: 2
- subtitulo con idIdioma = 2: 0
- subtitulo con idIdioma = 10: 2

DELETE FROM idioma WHERE idIdioma=10;
DELETE 1
RESULTADO:
- titulo_idioma con idIdioma = 10: 0
- subtitulo con idIdioma = 10: 0
```

Como podemos ver una actualización o el borrado afectaría a las tablas que referencia esta tabla,**idioma.**

```
====================
TABLA TITULO_IDIOMA:
====================

Insertamos:
-----------
INSERT INTO titulo_idioma (idTitulo, idIdioma) VALUES (1,2);
INSERT 0 1

 idtitulo | ididioma
----------+----------
        1 |        1
        2 |        1
        2 |        2
        3 |        1
        3 |        2
        4 |        1
        1 |        2
(7 rows)

Actualizamos:
-------------
UPDATE titulo_idioma SET idTitulo=4 WHERE idTitulo=1 AND idIdioma=2;
UPDATE 1

 idtitulo | ididioma
----------+----------
        1 |        1
        2 |        1
        2 |        2
        3 |        1
        3 |        2
        4 |        1
        4 |        2
(7 rows)

Eliminamos:
-----------
DELETE FROM titulo_idioma WHERE idTitulo=4 AND idIdioma=2;
DELETE 1

 idtitulo | ididioma
----------+----------
        1 |        1
        2 |        1
        2 |        2
        3 |        1
        3 |        2
        4 |        1
(6 rows)
```

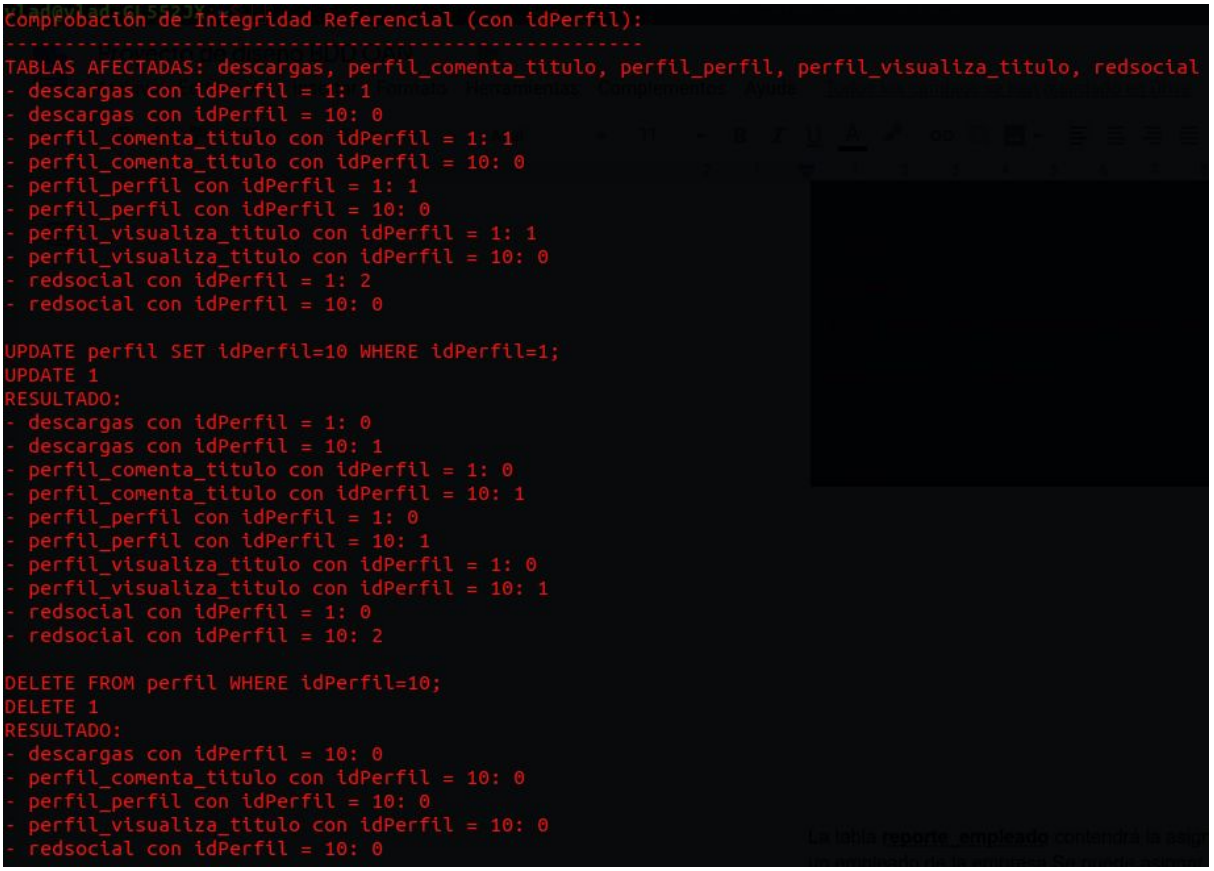
La tabla **titulo_idioma** contiene la información de la relación entre un título y el idioma que tendria este titulo.Es decir un titulo identificado por **idtitulo** se encuentra disponible en uno o múltiples idiomas **ididioma** referenciados de la tabla **idioma.**



```
=============
TABLA PERFIL:
=============

Insertamos:
-----------
INSERT INTO perfil (nombre, email) VALUES (Prueba, antonio.gutierrez1984@gmail.com);
INSERT 0 1

 idperfil | nombre  | num_descargas_actuales |             email              | idpendientes | idfavoritos
----------+---------+------------------------+--------------------------------+--------------+-------------
        3 | Julia   |                      0 | julia.almeida@gmail.com        |            3 |           3
        1 | Antonio |                      1 | antonio.gutierrez1984@gmail.com |            1 |           1
        2 | Toni    |                      1 | antonio.gutierrez1984@gmail.com |            2 |           2
        4 | Vicky   |                      1 | victoria.md@gmail.com          |            4 |           4
        5 | Prueba  |                      0 | antonio.gutierrez1984@gmail.com |            5 |           5
(5 rows)

Actualizamos:
-------------
UPDATE perfil
SET nombre=PruebaActualizado
WHERE nombre=Prueba AND email=antonio.gutierrez1984@gmail.com;
UPDATE 1

 idperfil |      nombre       | num_descargas_actuales |             email              | idpendientes | idfavoritos
----------+-------------------+------------------------+--------------------------------+--------------+-------------
        3 | Julia             |                      0 | julia.almeida@gmail.com        |            3 |           3
        1 | Antonio           |                      1 | antonio.gutierrez1984@gmail.com |            1 |           1
        2 | Toni              |                      1 | antonio.gutierrez1984@gmail.com |            2 |           2
        4 | Vicky             |                      1 | victoria.md@gmail.com          |            4 |           4
        5 | PruebaActualizado |                      0 | antonio.gutierrez1984@gmail.com |            5 |           5
(5 rows)

Eliminamos:
-----------
DELETE FROM perfil WHERE nombre=PruebaActualizado AND email=antonio.gutierrez1984@gmail.com;
DELETE 1

 idperfil | nombre  | num_descargas_actuales |             email              | idpendientes | idfavoritos
----------+---------+------------------------+--------------------------------+--------------+-------------
        3 | Julia   |                      0 | julia.almeida@gmail.com        |            3 |           3
        1 | Antonio |                      1 | antonio.gutierrez1984@gmail.com |            1 |           1
        2 | Toni    |                      1 | antonio.gutierrez1984@gmail.com |            2 |           2
        4 | Vicky   |                      1 | victoria.md@gmail.com          |            4 |           4
(4 rows)
```

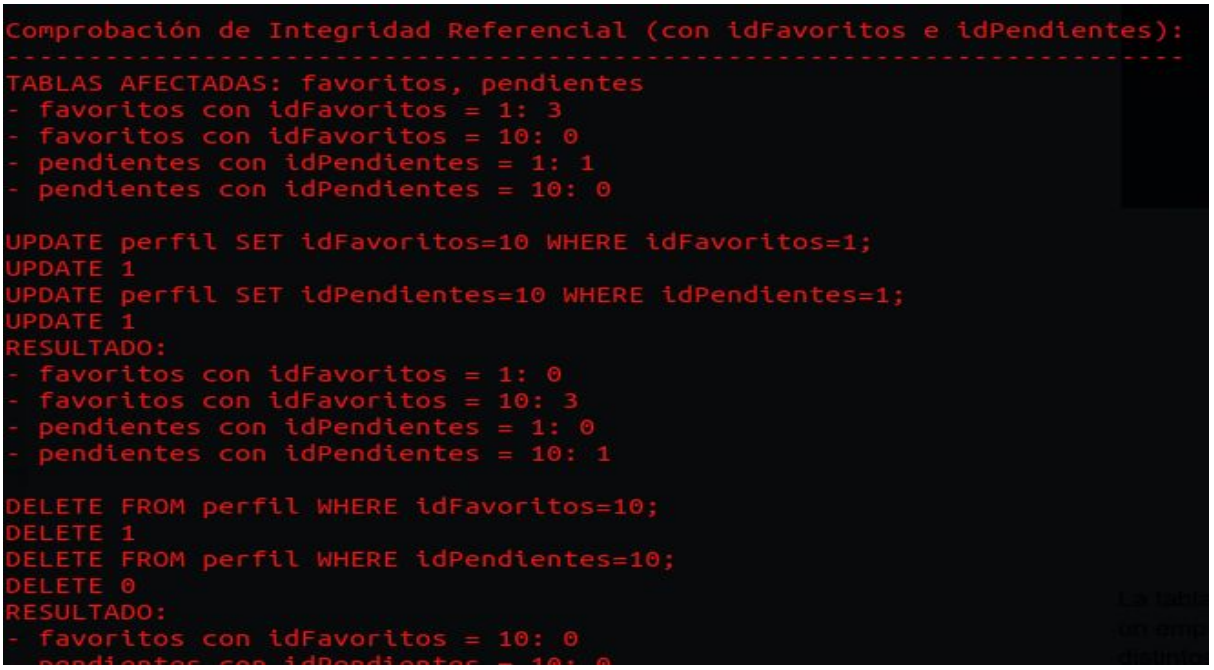
En la tabla **perfil** se definirá el perfil de un usuario,cada usuario puede tener varios perfiles dependiendo de su suscripción, y estos perfiles contendrá un nombre, el número de descargas actuales(atributo calculado en función de las descargas del perfil), el correo al que pertenezca el perfil y una lista de títulos pendientes y favoritos.

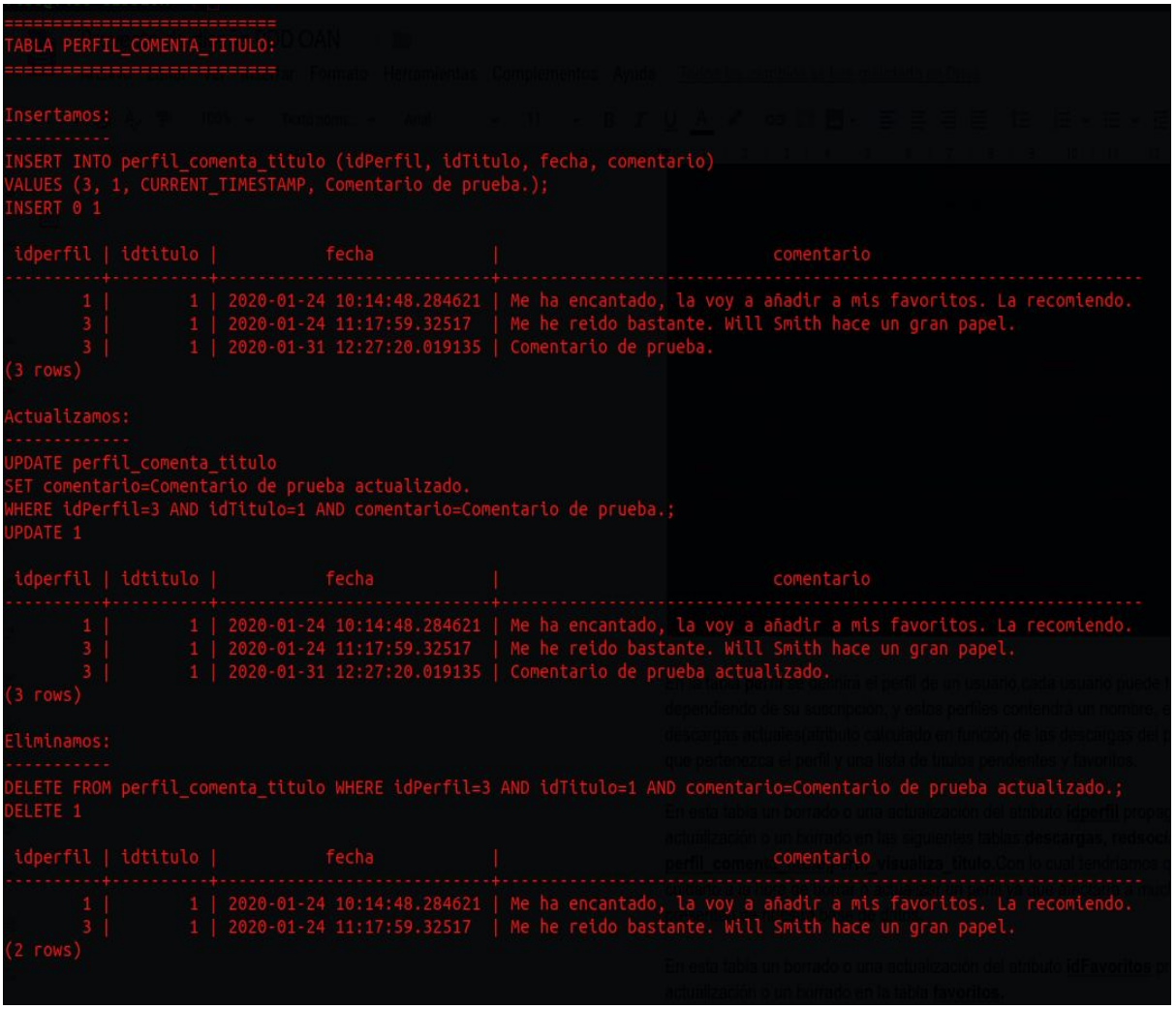
En esta tabla un borrado o una actualización del atributo **idperfil** propagara una actualización o un borrado en las siguientes tablas:**descargas, redsocial, perfil_perfil, perfil_comenta_titulo,perfil_visualiza_titulo.**Con lo cual tendríamos que tener mucho cuidado a la hora de borrar o actualizar un perfil ya que afectaría a muchas de las tablas presenten de nuestra base de datos.



En esta tabla un borrado o una actualización del atributo **idFavoritos** propagara una actualización o un borrado en la tabla **favoritos.**

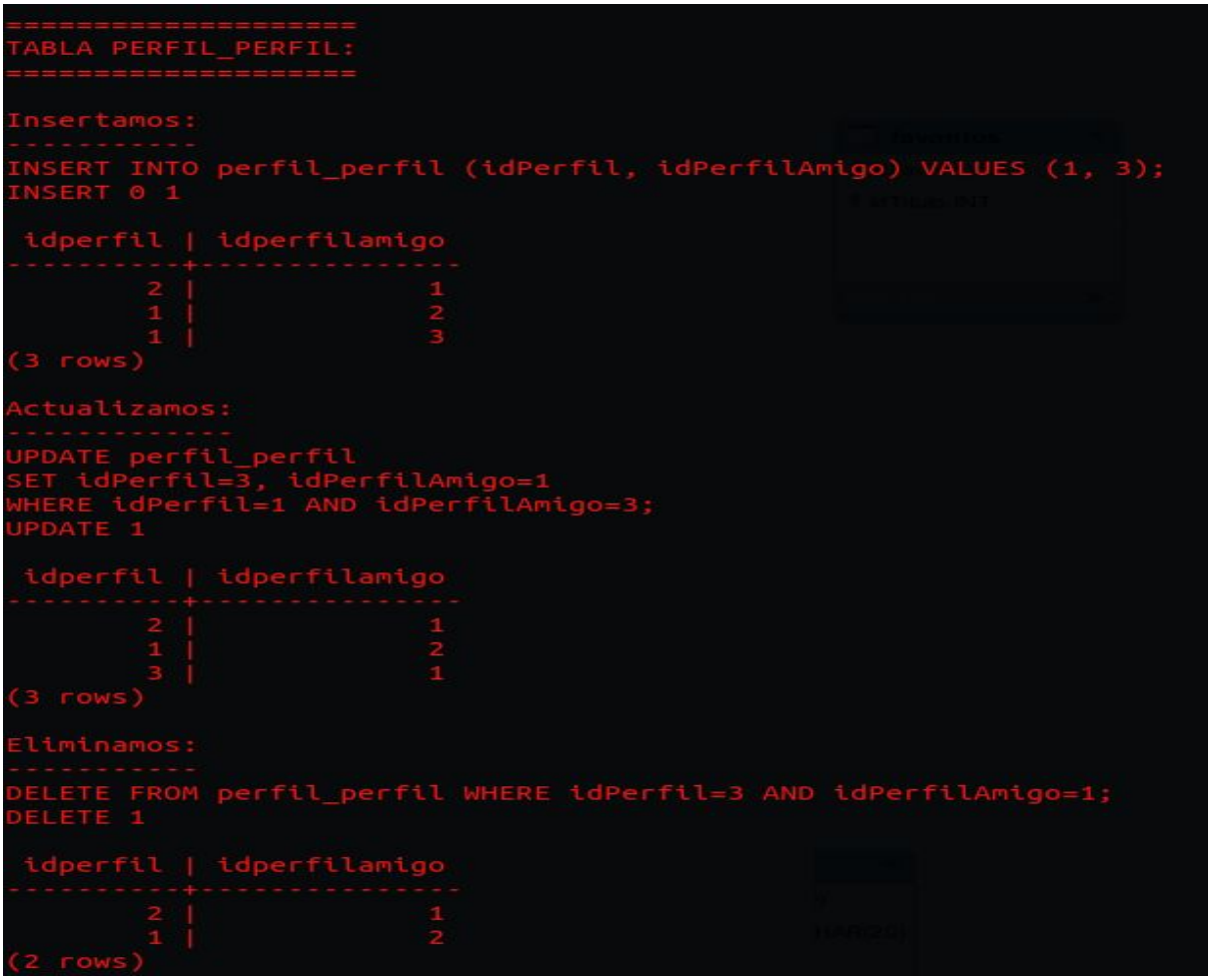
En esta tabla un borrado o una actualización del atributo **idpendientes** propagara una actualización o un borrado en la tabla **pendientes.**



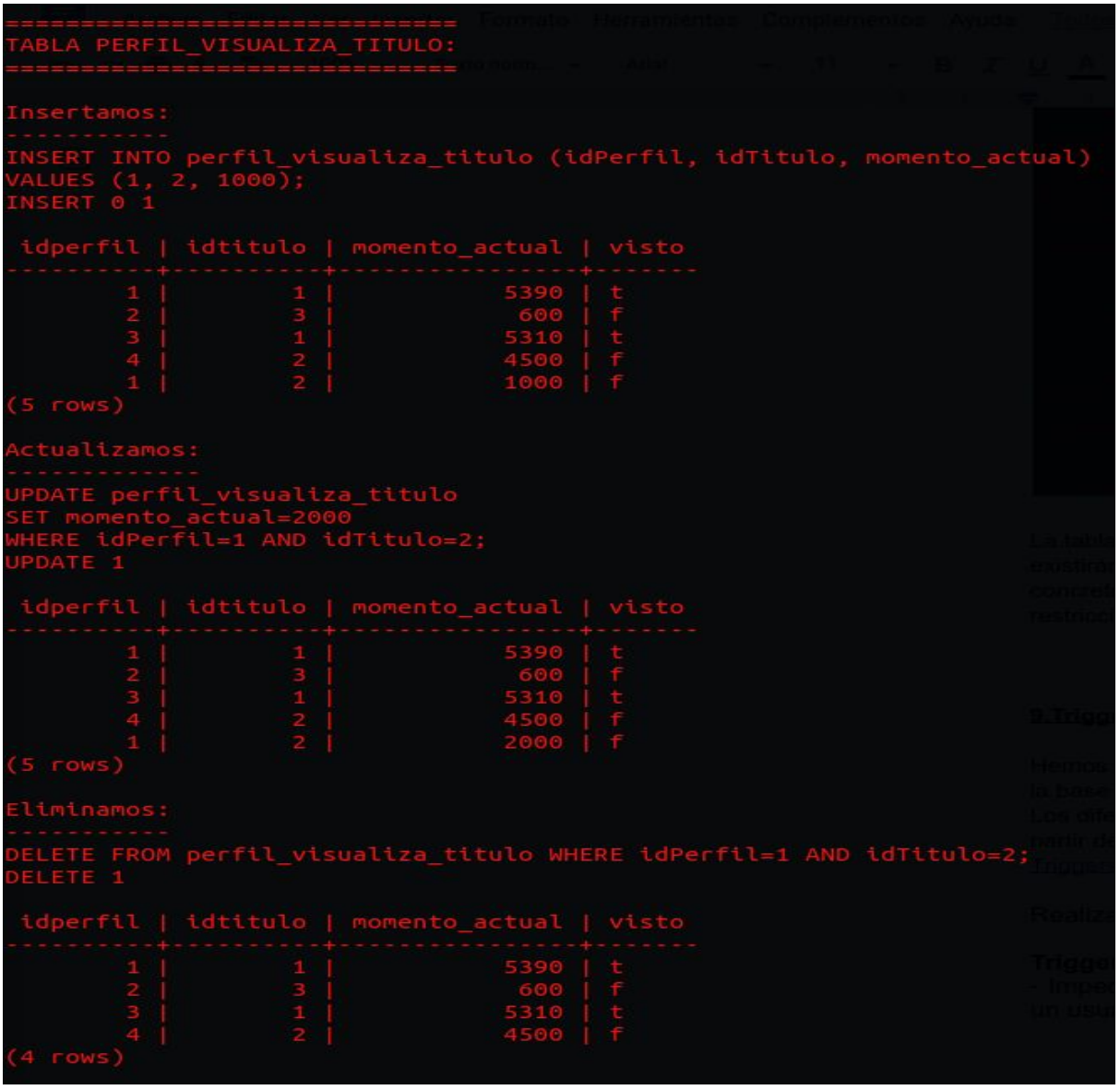


```
============================
TABLA PERFIL_COMENTA_TITULO:
============================

Insertamos:
-----------
INSERT INTO perfil_comenta_titulo (idPerfil, idTitulo, fecha, comentario)
VALUES (3, 1, CURRENT_TIMESTAMP, Comentario de prueba.);
INSERT 0 1

 idperfil | idtitulo |           fecha            |                          comentario
----------+----------+----------------------------+-------------------------------------------------------------
        1 |        1 | 2020-01-24 10:14:48.284621 | Me ha encantado, la voy a añadir a mis favoritos. La recomiendo.
        3 |        1 | 2020-01-24 11:17:59.32517  | Me he reido bastante. Will Smith hace un gran papel.
        3 |        1 | 2020-01-31 12:27:20.019135 | Comentario de prueba.
(3 rows)

Actualizamos:
-------------
UPDATE perfil_comenta_titulo
SET comentario=Comentario de prueba actualizado.
WHERE idPerfil=3 AND idTitulo=1 AND comentario=Comentario de prueba.;
UPDATE 1

 idperfil | idtitulo |           fecha            |                          comentario
----------+----------+----------------------------+-------------------------------------------------------------
        1 |        1 | 2020-01-24 10:14:48.284621 | Me ha encantado, la voy a añadir a mis favoritos. La recomiendo.
        3 |        1 | 2020-01-24 11:17:59.32517  | Me he reido bastante. Will Smith hace un gran papel.
        3 |        1 | 2020-01-31 12:27:20.019135 | Comentario de prueba actualizado.
(3 rows)

Eliminamos:
-----------
DELETE FROM perfil_comenta_titulo WHERE idPerfil=3 AND idTitulo=1 AND comentario=Comentario de prueba actualizado.;
DELETE 1

 idperfil | idtitulo |           fecha            |                          comentario
----------+----------+----------------------------+-------------------------------------------------------------
        1 |        1 | 2020-01-24 10:14:48.284621 | Me ha encantado, la voy a añadir a mis favoritos. La recomiendo.
        3 |        1 | 2020-01-24 11:17:59.32517  | Me he reido bastante. Will Smith hace un gran papel.
(2 rows)
```

La tabla **perfil_comenta_titulo** contendra la informacion necesaria para referenciar los comentarios hechos por un perfil de un cierto usuario a un cierto título.Por lo tanto, contendrá el id del perfil del usuario que haya hecho el comentario,el id del título sobre el cual se ha hecho, la fecha y el comentario en sí.Un perfil de usuario puede hacer varios comentarios sobre el mismo título.Una restricción que le hemos puesto es que es obligatorio que un perfil haya visto el título antes de poder comentarlo.

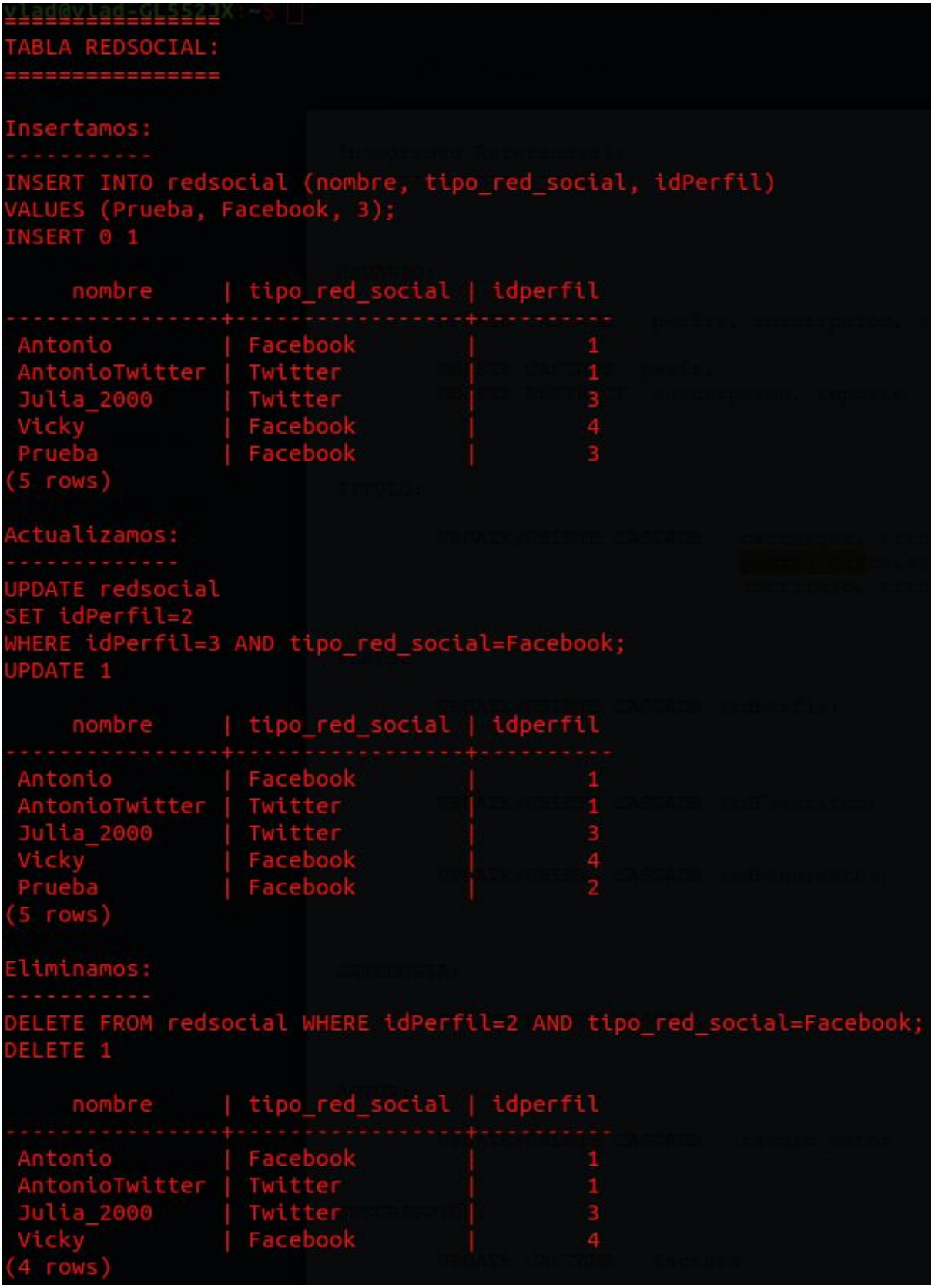


La tabla **perfil_perfil** contendrá la información sobre las “relaciones de amistad” que existirán entre los diferentes perfiles.Es decir,que otros perfiles serán amigos de un perfil concreto.También existe la restricción de que un perfil no puede ser amigo de sí mismo.Esta restricción se podrá comprobar en el siguiente apartado.



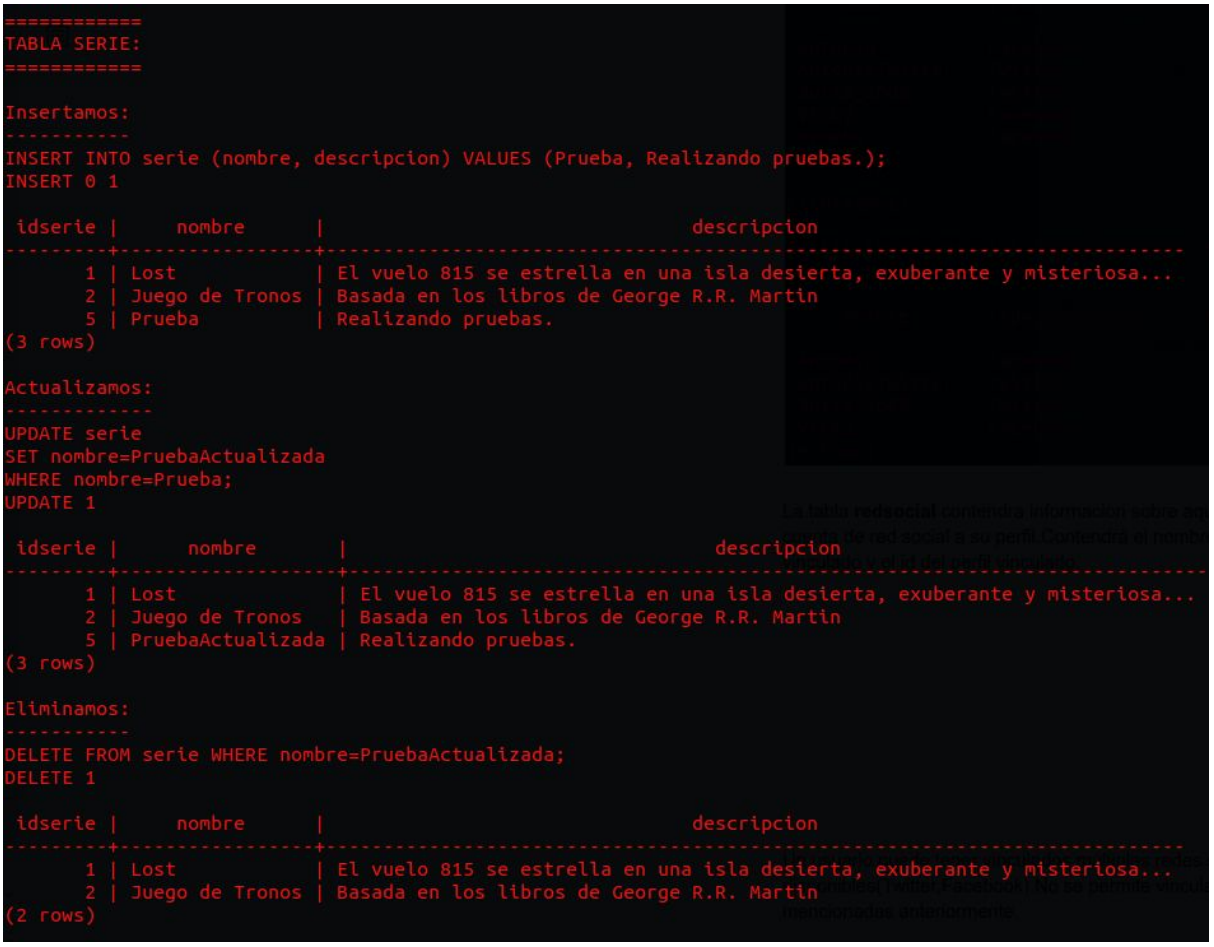
```
==============================
TABLA PERFIL_VISUALIZA_TITULO:
==============================

Insertamos:
-----------
INSERT INTO perfil_visualiza_titulo (idPerfil, idTitulo, momento_actual)
VALUES (1, 2, 1000);
INSERT 0 1

 idperfil | idtitulo | momento_actual | visto
----------+----------+----------------+-------
        1 |        1 |           5390 | t
        2 |        3 |            600 | f
        3 |        1 |           5310 | t
        4 |        2 |           4500 | f
        1 |        2 |           1000 | f
(5 rows)

Actualizamos:
-------------
UPDATE perfil_visualiza_titulo
SET momento_actual=2000
WHERE idPerfil=1 AND idTitulo=2;
UPDATE 1

 idperfil | idtitulo | momento_actual | visto
----------+----------+----------------+-------
        1 |        1 |           5390 | t
        2 |        3 |            600 | f
        3 |        1 |           5310 | t
        4 |        2 |           4500 | f
        1 |        2 |           2000 | f
(5 rows)

Eliminamos:
-----------
DELETE FROM perfil_visualiza_titulo WHERE idPerfil=1 AND idTitulo=2;
DELETE 1

 idperfil | idtitulo | momento_actual | visto
----------+----------+----------------+-------
        1 |        1 |           5390 | t
        2 |        3 |            600 | f
        3 |        1 |           5310 | t
        4 |        2 |           4500 | f
(4 rows)
```

La tabla **perfil_visualiza_titulo** contendrá aquella información de los títulos que está visualizando o haya visualizado un perfil de un usuario.Por lo tanto,contendrá información sobre el id del perfil que está viendo el título, el id del título que se está viendo, el momento actual del título(en segundos totales) y se se ha visto completamente o no.



La tabla **redsocial** contendra informacion sobre aquellos perfiles que hayan vinculado su cuenta de red social a su perfil.Contendrá el nombre visible,tipo de red social que se ha vinculado y el id del perfil vinculado.

Un usuario puede tener vinculados múltiples redes sociales pero de las que estan disponibles(Twitter,Facebook).No se permite vincular a otra red social a parte de las mencionadas anteriormente.



```
============
TABLA SERIE:
============

Insertamos:
-----------
INSERT INTO serie (nombre, descripcion) VALUES (Prueba, Realizando pruebas.);
INSERT 0 1

 idserie |     nombre      |                              descripcion
---------+-----------------+-----------------------------------------------------------------------
       1 | Lost            | El vuelo 815 se estrella en una isla desierta, exuberante y misteriosa...
       2 | Juego de Tronos | Basada en los libros de George R.R. Martin
       5 | Prueba          | Realizando pruebas.
(3 rows)

Actualizamos:
-------------
UPDATE serie
SET nombre=PruebaActualizada
WHERE nombre=Prueba;
UPDATE 1

 idserie |      nombre       |                               descripcion
---------+-------------------+-----------------------------------------------------------------------
       1 | Lost              | El vuelo 815 se estrella en una isla desierta, exuberante y misteriosa...
       2 | Juego de Tronos   | Basada en los libros de George R.R. Martin
       5 | PruebaActualizada | Realizando pruebas.
(3 rows)

Eliminamos:
-----------
DELETE FROM serie WHERE nombre=PruebaActualizada;
DELETE 1

 idserie |     nombre      |                              descripcion
---------+-----------------+-----------------------------------------------------------------------
       1 | Lost            | El vuelo 815 se estrella en una isla desierta, exuberante y misteriosa...
       2 | Juego de Tronos | Basada en los libros de George R.R. Martin
(2 rows)
```

La tabla **serie** contendrá información sobre aquellos títulos que son de tipo Serie, cada serie tendrá un id único que le permitirá identificar, un nombre y una descripción.

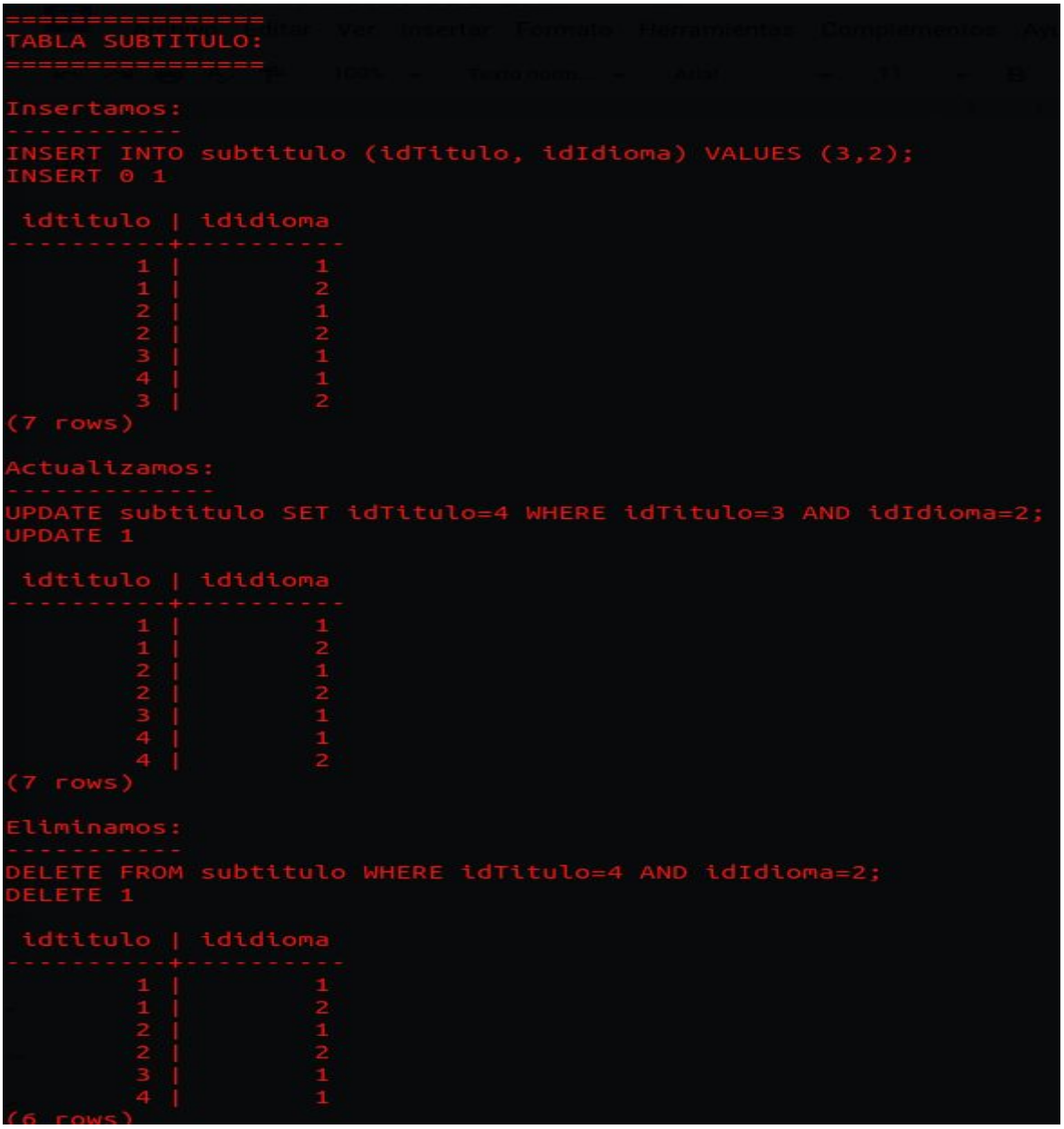
```
====================
TABLA TITULO_SERIE:
====================

Insertamos:
-----------
INSERT INTO titulo_serie (idTitulo, capitulo, temporada, idSerie) VALUES (4, 2, 3, 1);
INSERT 0 1

 idtitulo | capitulo | temporada | idserie
----------+----------+-----------+---------
        3 |        1 |         3 |       1
        4 |        2 |         3 |       1
(2 rows)

Actualizamos:
-------------
UPDATE titulo_serie SET capitulo=10 WHERE idTitulo=4;
UPDATE 1

 idtitulo | capitulo | temporada | idserie
----------+----------+-----------+---------
        3 |        1 |         3 |       1
        4 |       10 |         3 |       1
(2 rows)

Eliminamos:
-----------
DELETE FROM titulo_serie WHERE idTitulo=4;
DELETE 1

 idtitulo | capitulo | temporada | idserie
----------+----------+-----------+---------
        3 |        1 |         3 |       1
(1 row)
```

En la tabla **titulo_serie** se encuentra la información relevante a la relación entre la tabla **serie** y **titulo**, ya que almacena información como el identificador del titulo **idtitulo**, el identificador de la serie **idserie** y otra información como capitulo y temporada.
De esta manera se podría identificar fácilmente a qué serie pertenece cada título y además identificar que capitulo y de que temporada es.

Una actualización en la tabla **serie** propagara una actualización en cascada en la tabla **titulo_serie.**Sin embargo un borrado en la tabla **serie** está restriccionado y por lo tanto daría error en caso de que se produjera.

```
Comprobación de Integridad Referencial:
---------------------------------------
TABLAS AFECTADAS: titulo_serie
- titulo_serie con idSerie = 1: 1
- titulo_serie con idSerie = 10: 0

UPDATE serie SET idSerie=10 WHERE idSerie=1;
UPDATE 1
RESULTADO:
- titulo_serie con idSerie = 1: 0
- titulo_serie con idSerie = 10: 1

DELETE FROM serie WHERE idSerie=10;
psql:/home/vlad/Desktop/AyDBDD/OAN/pruebas_insercion.sql:1103: ERROR:  update or delete on table "serie"
DETAIL:  Key (idserie)=(10) is still referenced from table "titulo_serie".
```

Como podemos ver las actualizaciones están permitidas pero el borrado no.

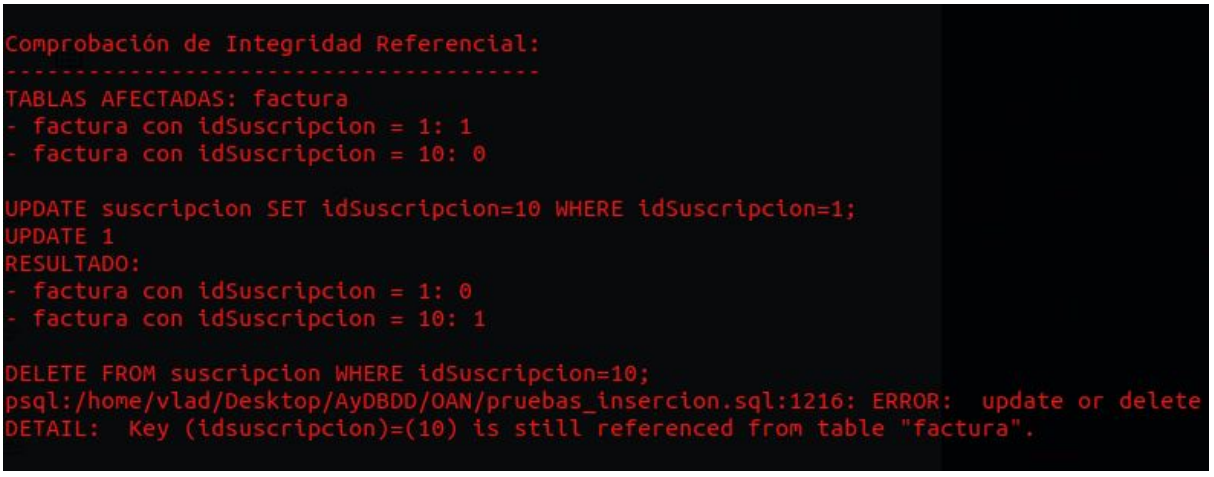


La tabla **subtitulo** referencia a aquellos títulos que estaran subtitulos en alguna idioma.Es decir el titulo **idtitulo** estara subtitulado en los idiomas referenciados por **ididioma**.
Un título puede estar subtitulado en multiples idiomas.

```
==================
TABLA SUSCRIPCION:
==================

Insertamos:
-----------
INSERT INTO suscripcion (fecha_finalizacion, num_descargas_max, tipo_suscripcion, email)
VALUES (2019-01-01, 100, Premium, antonio.gutierrez1984@gmail.com);
INSERT 0 1

 idsuscripcion | fecha_finalizacion | num_descargas_max | tipo_suscripcion |              email
---------------+--------------------+-------------------+------------------+---------------------------------
             1 | 2020-01-01         |                 2 | Premium          | antonio.gutierrez1984@gmail.com
             2 | 2021-01-01         |                 2 | Premium          | antonio.gutierrez1984@gmail.com
             3 | 2020-06-01         |                 0 | Basica           | julia.almeida@gmail.com
             4 | 2020-05-30         |                 4 | Contenido        | victoria.md@gmail.com
             7 | 2019-01-01         |               100 | Premium          | antonio.gutierrez1984@gmail.com
(5 rows)

Actualizamos:
-------------
UPDATE suscripcion
SET num_descargas_max=50
WHERE email=antonio.gutierrez1984@gmail.com AND fecha_finalizacion=2019-01-01;
UPDATE 1

 idsuscripcion | fecha_finalizacion | num_descargas_max | tipo_suscripcion |              email
---------------+--------------------+-------------------+------------------+---------------------------------
             1 | 2020-01-01         |                 2 | Premium          | antonio.gutierrez1984@gmail.com
             2 | 2021-01-01         |                 2 | Premium          | antonio.gutierrez1984@gmail.com
             3 | 2020-06-01         |                 0 | Basica           | julia.almeida@gmail.com
             4 | 2020-05-30         |                 4 | Contenido        | victoria.md@gmail.com
             7 | 2019-01-01         |                50 | Premium          | antonio.gutierrez1984@gmail.com
(5 rows)

Eliminamos:
-----------
DELETE FROM suscripcion WHERE email=antonio.gutierrez1984@gmail.com AND fecha_finalizacion=2019-01-01;
DELETE 1

 idsuscripcion | fecha_finalizacion | num_descargas_max | tipo_suscripcion |              email
---------------+--------------------+-------------------+------------------+---------------------------------
             1 | 2020-01-01         |                 2 | Premium          | antonio.gutierrez1984@gmail.com
             2 | 2021-01-01         |                 2 | Premium          | antonio.gutierrez1984@gmail.com
             3 | 2020-06-01         |                 0 | Basica           | julia.almeida@gmail.com
             4 | 2020-05-30         |                 4 | Contenido        | victoria.md@gmail.com
(4 rows)
```

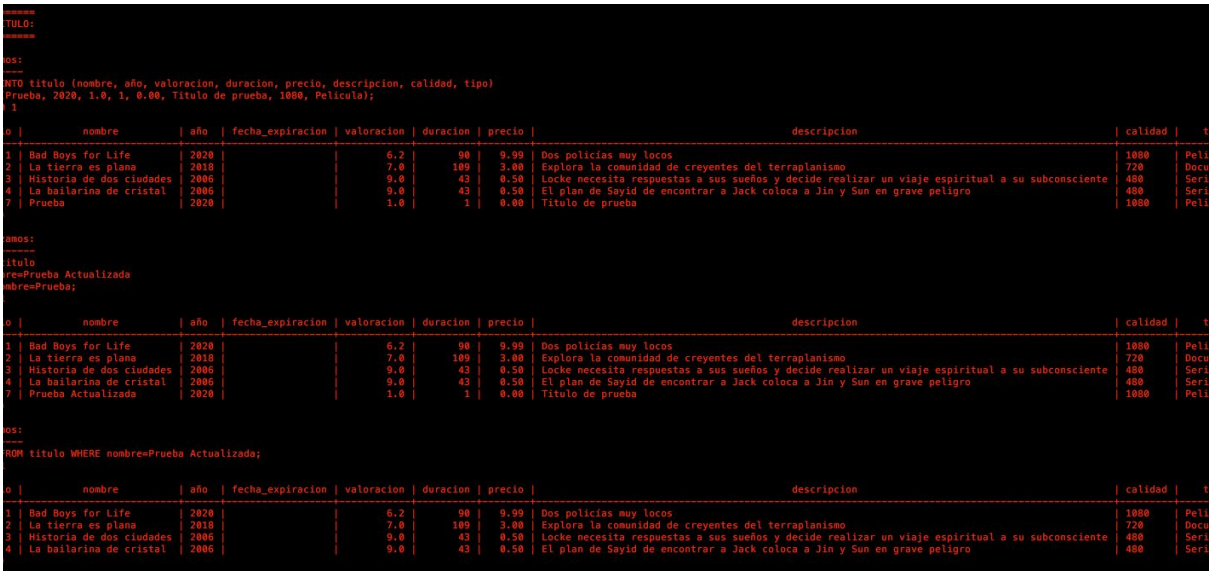
La tabla **suscripcion** contendra la informacion de las distintas suscripciones de los usuarios identificados por su correo.Contendrá un identificador único de la suscripción,fecha de finalización, número de descargas máximas permitidas y el tipo de suscripción que se haya hecho.

Una actualización en la tabla **suscripcion** propagara una actualización en cascada en la tabla **factura.**Sin embargo un borrado en la tabla **suscripcion** está restriccionado y por lo tanto daría error en caso de que se produjera.



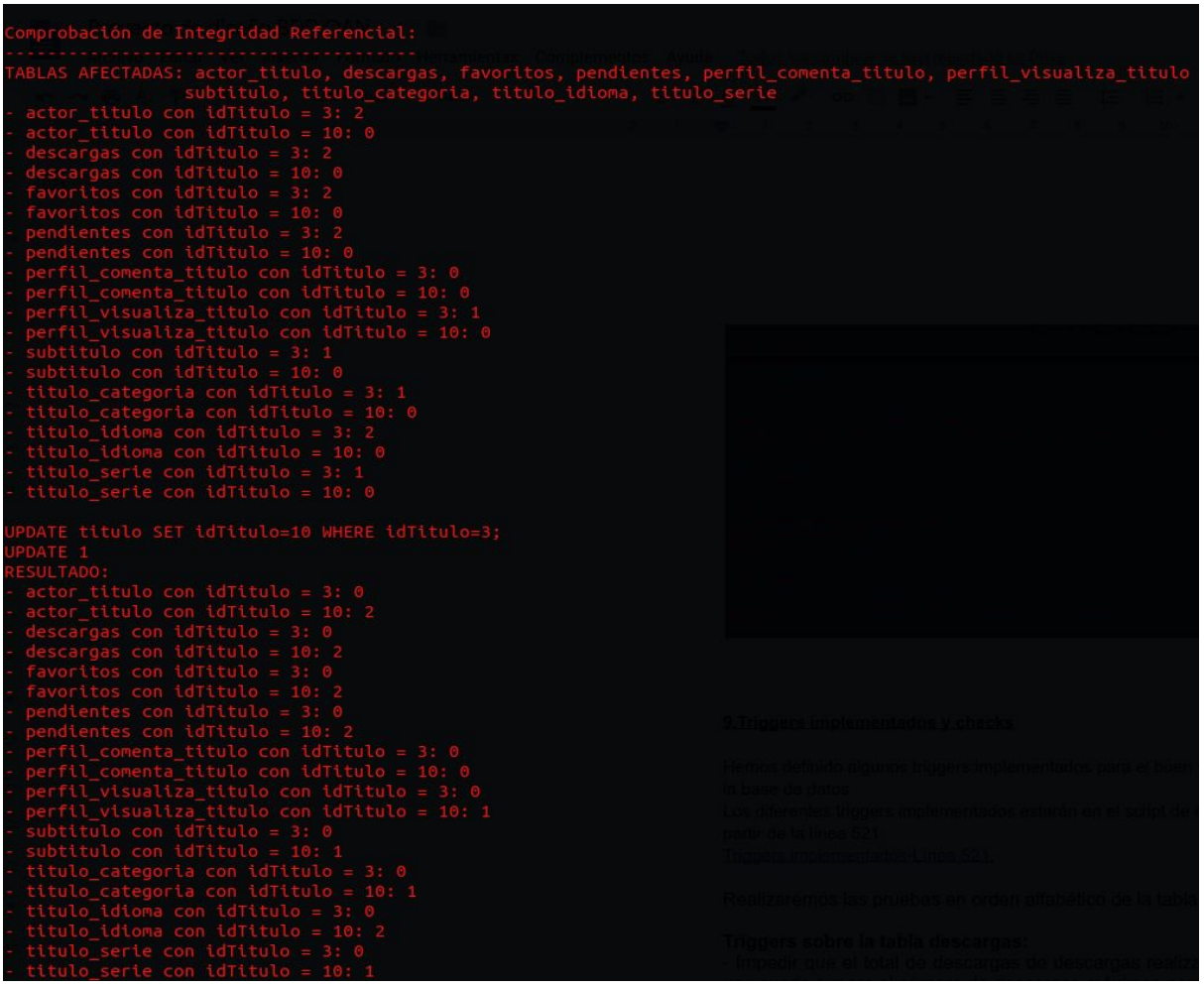
```
Comprobación de Integridad Referencial:
---------------------------------------
TABLAS AFECTADAS: factura
- factura con idSuscripcion = 1: 1
- factura con idSuscripcion = 10: 0

UPDATE suscripcion SET idSuscripcion=10 WHERE idSuscripcion=1;
UPDATE 1
RESULTADO:
- factura con idSuscripcion = 1: 0
- factura con idSuscripcion = 10: 1

DELETE FROM suscripcion WHERE idSuscripcion=10;
psql:/home/vlad/Desktop/AyDBDD/OAN/pruebas_insercion.sql:1216: ERROR:  update or delete
DETAIL:  Key (idsuscripcion)=(10) is still referenced from table "factura".
```

Como podemos ver las actualizaciones están permitidas pero el borrado no.

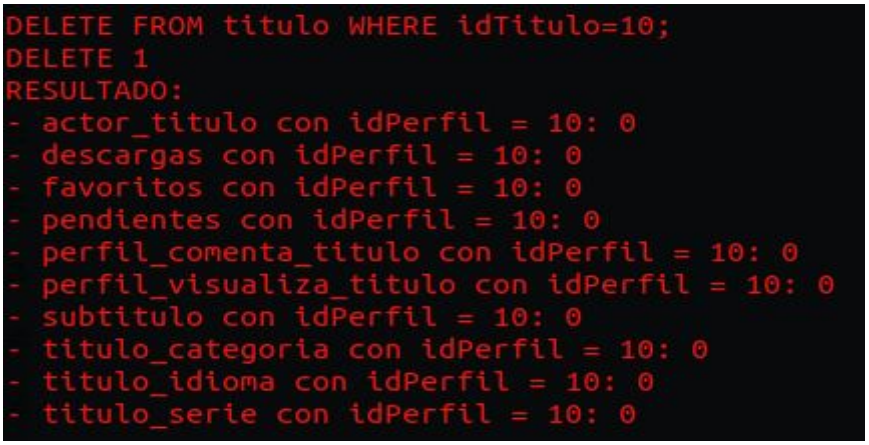


La tabla **titulo** contendra la informacion relevante en referencia a los distintos títulos que presenta la plataforma.Tiene un id único que lo identifica,un nombre,un año,una fecha de expiración,una valoración media,una duración,un precio(en caso de que el usuario tenga una suscripción por contenido),una descripción,calidad(calidad de reproducción),y un tipo que puede ser:película,serie,documental.

En esta tabla un borrado o una actualización propagara una actualización o un borrado en las siguientes tablas:**descargas, titulo_categoria, actor_titulo, perfil_comenta_titulo, perfil_visualiza_titulo, pendientes, favoritos, titulo_serie,subtitulo, titulo_idioma.** Con lo cual tendríamos que tener mucho cuidado a la hora de borrar o actualizar un **titulo** ya que afectaría a muchas de las tablas presenten de nuestra base de datos.



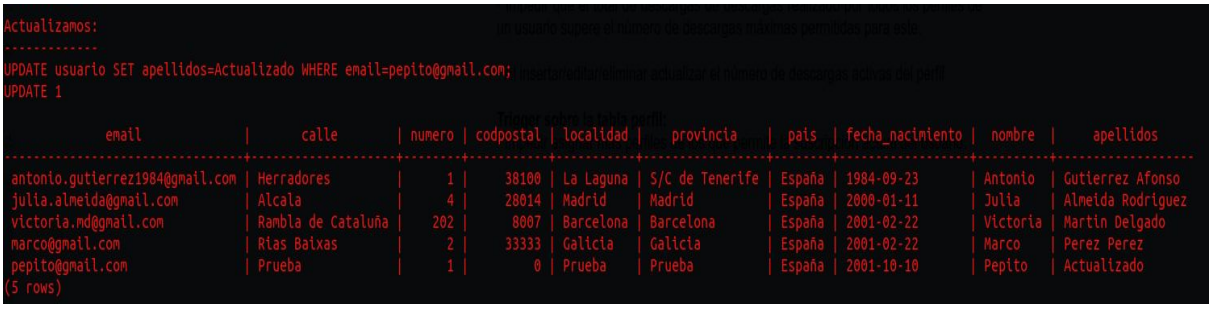
Como podemos ver en caso de una actualización se verían afectadas todas las tablas mencionadas anteriormente,tablas que tendrían como clave ajena la clave primaria de la tabla principal **titulo:idtitulo.**



Por otra parte está el borrado donde podremos ver que se propaga en cascada en todas las tablas “hijas” a las que referencia la tabla **titulo.**

Por último,la tabla **usuario** contendrá la información relevante a cada usuario de la plataforma que se quiere suscribir al servicio de OAN.Para ello es necesario que se identifique con un correo(clave primaria),que contenga una dirección formada por nombre de la calle,número,código postal,localidad,provincia y país.También debe cumplimentar con la fecha de nacimiento para asegurar que ciertos usuarios no puedan ver cierto tipo de títulos y por último deben completar su nombre y apellido.

```
==============
TABLA USUARIO:
==============

Insertamos:
-----------
INSERT INTO usuario (email, calle, numero, codPostal, localidad, provincia, pais, fecha_nacimiento, nombre, apellidos)
VALUES (pepito@gmail.com, Prueba, 1, 00000, Prueba, Prueba, España, 2001-10-10, Pepito, de Prueba);
INSERT 0 1

            email             |      calle        | numero | codpostal | localidad |    provincia    |  pais  | fecha_nacimiento |  nombre  |     apellidos
------------------------------+-------------------+--------+-----------+-----------+-----------------+--------+------------------+----------+-------------------
 antonio.gutierrez1984@gmail.com | Herradores        |      1 |     38100 | La Laguna | S/C de Tenerife | España | 1984-09-23       | Antonio  | Gutierrez Afonso
 julia.almeida@gmail.com         | Alcala            |      4 |     28014 | Madrid    | Madrid          | España | 2000-01-11       | Julia    | Almeida Rodriguez
 victoria.md@gmail.com           | Rambla de Cataluña |    202 |      8007 | Barcelona | Barcelona       | España | 2001-02-22       | Victoria | Martin Delgado
 marco@gmail.com                 | Rias Baixas       |      2 |     33333 | Galicia   | Galicia         | España | 2001-02-22       | Marco    | Perez Perez
 pepito@gmail.com                | Prueba            |      1 |         0 | Prueba    | Prueba          | España | 2001-10-10       | Pepito   | de Prueba
(5 rows)
```

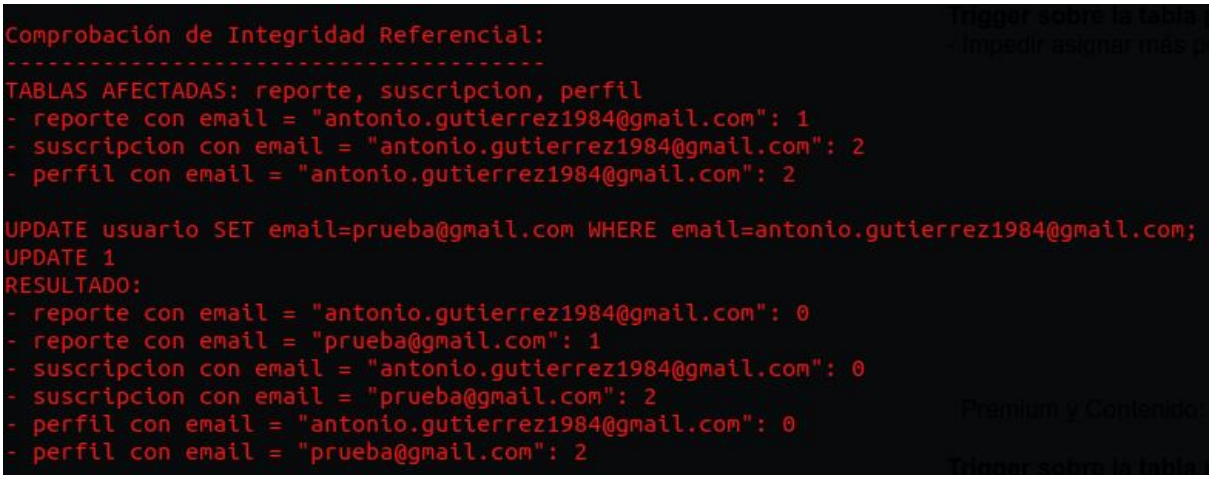


```
Actualizamos:
-------------
UPDATE usuario SET apellidos=Actualizado WHERE email=pepito@gmail.com;
UPDATE 1

            email             |      calle        | numero | codpostal | localidad |    provincia    |  pais  | fecha_nacimiento |  nombre  |     apellidos
------------------------------+-------------------+--------+-----------+-----------+-----------------+--------+------------------+----------+-------------------
 antonio.gutierrez1984@gmail.com | Herradores        |      1 |     38100 | La Laguna | S/C de Tenerife | España | 1984-09-23       | Antonio  | Gutierrez Afonso
 julia.almeida@gmail.com         | Alcala            |      4 |     28014 | Madrid    | Madrid          | España | 2000-01-11       | Julia    | Almeida Rodriguez
 victoria.md@gmail.com           | Rambla de Cataluña |    202 |      8007 | Barcelona | Barcelona       | España | 2001-02-22       | Victoria | Martin Delgado
 marco@gmail.com                 | Rias Baixas       |      2 |     33333 | Galicia   | Galicia         | España | 2001-02-22       | Marco    | Perez Perez
 pepito@gmail.com                | Prueba            |      1 |         0 | Prueba    | Prueba          | España | 2001-10-10       | Pepito   | Actualizado
(5 rows)
```

```
Eliminamos:
-----------
DELETE FROM usuario WHERE email=pepito@gmail.com;
DELETE 1

            email             |      calle        | numero | codpostal | localidad |    provincia    |  pais  | fecha_nacimiento |  nombre  |     apellidos
------------------------------+-------------------+--------+-----------+-----------+-----------------+--------+------------------+----------+-------------------
 antonio.gutierrez1984@gmail.com | Herradores        |      1 |     38100 | La Laguna | S/C de Tenerife | España | 1984-09-23       | Antonio  | Gutierrez Afonso
 julia.almeida@gmail.com         | Alcala            |      4 |     28014 | Madrid    | Madrid          | España | 2000-01-11       | Julia    | Almeida Rodriguez
 victoria.md@gmail.com           | Rambla de Cataluña |    202 |      8007 | Barcelona | Barcelona       | España | 2001-02-22       | Victoria | Martin Delgado
 marco@gmail.com                 | Rias Baixas       |      2 |     33333 | Galicia   | Galicia         | España | 2001-02-22       | Marco    | Perez Perez
(4 rows)
```

No habrá restricciones para la modificación,actualización o inserción de nuevos datos en esta tabla.

En esta tabla una actualización propagara una actualización en las siguientes tablas:**perfil,suscripcion,reporte.**
El borrado de esta tabla propagara un borrado en la siguiente tabla:**perfil.**
Asimismo el borrado estará restricciones y no se propagara en las tablas:**suscripcion,reporte.**



Como podemos ver la integridad referencial en caso de la modificación de la tabla **usuario** se vería afectada en las tablas **reporte,suscripcion,perfil** con lo cual las modificaciones en la tabla padre **usuario** se estaran propagando en las tablas hijas.

Como antes hemos mencionado el borrado estará restricciones en las tablas **reporte,suscripcion** y en caso de que se intente borrar un correo de la tabla padre **usuario** saltaría el error de integridad referencial.

Triggers implementados y checks

En esta sección implementaremos los diferentes triggers y comprobaremos su correcto funcionamiento.

Hemos definido algunos triggers implementados para el buen y correcto funcionamiento de la base de datos.
Los diferentes triggers implementados estarán en el siguiente script:Triggers OAN

Realizaremos las pruebas en orden alfabético de la tabla que es utilizada.

Triggers sobre la tabla descargas:
- Impedir que el total de descargas de descargas realizado por todos los perfiles de un usuario supere el número de descargas máximas permitidas para este.

- Al insertar/editar/eliminar actualizar el número de descargas activas del perfil

Trigger sobre la tabla perfil:
- Impedir asignar más perfiles de los que permite la suscripción actual del usuario.
  Premium y Contenido: 4. Básica: 1

Trigger sobre la tabla perfil_comenta_titulo:
- Impedir que se comente un título que no haya sido visualizado

Trigger sobre la tabla perfil_perfil:
- El perfil no puede ser amigo de sí mismo

Triggers sobre la tabla perfil_visualiza_titulo:
- Al insertar/modificar un registro comprueba si el momento_actual coincide aproximadamente con la duración del título. Si es así, lo marca como visto.

- Al marcar un título como visto, si está en descargas del usuario, eliminarlo.

Trigger sobre la tabla suscripción:
- Impedir que un usuario tenga más de 1 suscripción activa

Trigger sobre la tabla título:
- Impedir la modificación del tipo de un título, si anteriormente era serie y aún sigue asociada a una serie.

Trigger sobre la tabla título_serie:
- Comprobar que el título que se desea asociar a una serie es del tipo serie.
Las pruebas realizadas se hicieron con el propósito de comprobar el correcto funcionamiento de los distintos triggers y se realizarán en el orden descrito anteriormente.Es importante que se cargue el fichero de datos y estos datos no se modifiquen.
Pruebas sobre el funcionamiento de los triggers de la base de datos

En la primera prueba:

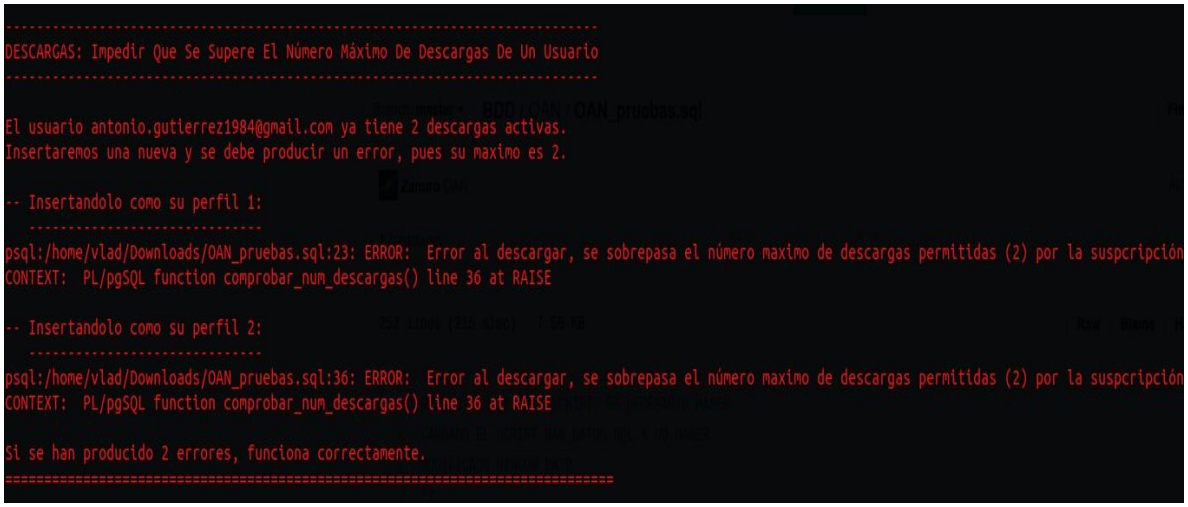
Tenemos el usuario antonio.gutierrez1984@gmail.com con 2 descargas activas.Cada cuenta tiene un número máximo de 2 descargas independientemente del número de perfiles que tenga.Por lo tanto si intentamos añadir otra descarga activa asociada a uno de los perfiles de esta cuenta nos debería dar un error:

Insertandolo con su perfil 1:

```
INSERT INTO descargas (idTitulo, fecha_caducidad, idPerfil)
VALUES (1, CURRENT_DATE + 1, 1);
```

Insertandolo con su perfil 2:

```
INSERT INTO descargas (idTitulo, fecha_caducidad, idPerfil)
VALUES (1, CURRENT_DATE + 1, 2);
```



Vemos como el trigger comprueba que el usuario no puede tener otra descarga activa y salta el trigger.

En la segunda prueba:

En esta prueba hemos probado insertar/actualizar/borrar el número de descargas activas de un perfil.

Obtener el número de descargas de un perfil:

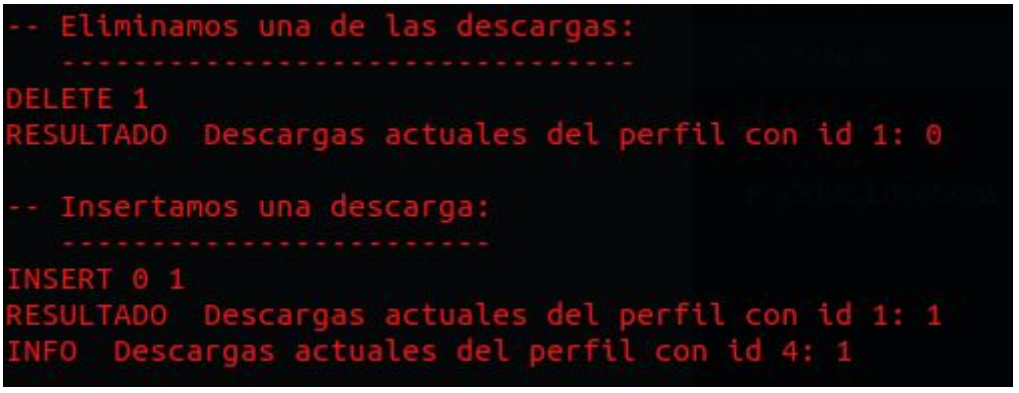
```
SELECT num_descargas_actuales AS ndescargas FROM perfil WHERE
idperfil=1
\echo 'INFO  Descargas actuales del perfil con id 1:' :ndescargas
```

Resultado:

Borrar o insertar descargas:

```
\echo '-- Eliminamos una de las descargas:'
\echo '   --------------------------------'
SELECT idTitulo AS titulo FROM descargas WHERE idperfil=1 LIMIT 1
\gset
DELETE FROM descargas WHERE idTitulo=:titulo AND idPerfil = 1;
SELECT num_descargas_actuales AS ndescargas FROM perfil WHERE
idperfil=1
\gset
\echo 'RESULTADO  Descargas actuales del perfil con id 1:' :ndescargas
\echo
\echo '-- Insertamos una descarga:'
\echo '   ------------------------'
INSERT INTO descargas (idTitulo, fecha_caducidad, idPerfil) VALUES (3,
CURRENT_DATE + 1, 1);
SELECT num_descargas_actuales AS ndescargas FROM perfil WHERE
idperfil=1
\gset
\echo 'RESULTADO  Descargas actuales del perfil con id 1:' :ndescargas
SELECT num_descargas_actuales AS ndescargas FROM perfil WHERE
idperfil=4
\gset
\echo 'INFO  Descargas actuales del perfil con id 4:' :ndescargas
```

Resultado:



Por último. probamos las actualización cambiando el perfil de la descarga:

```
\echo '-- Actualizamos una de las descargas, cambiandola de perfil:'
\echo '   ------------------------------------------------------'
SELECT idTitulo AS titulo FROM descargas WHERE idperfil=1 LIMIT 1
\gset
```

```
UPDATE descargas SET idPerfil=4 WHERE idTitulo=:titulo AND idPerfil=1;
SELECT num_descargas_actuales AS ndescargas FROM perfil WHERE idperfil=1
\gset
\echo 'RESULTADO  Descargas actuales del perfil con id 1:' :ndescargas
SELECT num_descargas_actuales AS ndescargas FROM perfil WHERE idperfil=4
\gset
\echo 'RESULTADO  Descargas actuales del perfil con id 4:' :ndescargas
\echo
```

Resultado:

```
-- Actualizamos una de las descargas, cambiandola de perfil:
   ------------------------------------------------------------
UPDATE 1
RESULTADO  Descargas actuales del perfil con id 1: 0
RESULTADO  Descargas actuales del perfil con id 4: 2
```

En la tercera prueba:
En esta prueba se impide que una cuenta tenga más perfiles asignados de los permitidos:Max. Premium y Contenido: 4. Básica: 1

```
\echo ' PERFIL: Impedir asignar más perfiles de los que permite la
suscripción actual del usuario.'

\echo 'INFO   El usuario "antonio.gutierrez1984@gmail.com" tiene una
suscripcion PREMIUM'
\echo '      y ya tiene 2 perfiles, su máximo son 4.'
\echo
\echo '-- Insertamos 2 nuevos perfiles:'
\echo '   ----------------------------'
INSERT   INTO   perfil   (nombre,   email)   VALUES   ('Antonio   2',
'antonio.gutierrez1984@gmail.com');
INSERT   INTO   perfil   (nombre,   email)   VALUES   ('Antonio   3',
'antonio.gutierrez1984@gmail.com');
\echo
\echo '-- Ahora, al intentar insertar otro, nos debe dar Error:'
\echo '   ------------------------------------------------------'
INSERT   INTO   perfil   (nombre,   email)   VALUES   ('Antonio   4',
'antonio.gutierrez1984@gmail.com');
```

Resultado:

```
INFO  El usuario "antonio.gutierrez1984@gmail.com" tiene una suscripcion PREMIUM
      y ya tiene 2 perfiles, su máximo son 4.

-- Insertamos 2 nuevos perfiles:
   --------------------------
INSERT 0 1
INSERT 0 1

-- Ahora, al intentar insertar otro, nos debe dar Error:
   ------------------------------------------------
psql:/home/vlad/Downloads/OAN_pruebas.sql:125: ERROR:  Error, Se excede del número máximo de perfiles (4) que permite tu suscripcion (Premium)
CONTEXT:  PL/pgSQL function comprobar_numero_perfiles() line 30 at RAISE
```

El usuario tenía 2 perfiles de un máximo de cuatro,se han introducido otros 2 y al intentar introducirse otros salta el trigger.

Otro ejemplo:

```
\echo 'INFO   El usuario "julia.almeida@gmail.com" tiene una suscripcion
BASICA'
 \echo ' y ya tiene 1 perfil, su máximo es 1.'
 \echo
 \echo '-- Al intentar insertar otro nos debe dar Error':
 \echo '   ---------------------------------------------'
 INSERT    INTO    perfil    (nombre,    email)    VALUES    ('Julia    2',
 'julia.almeida@gmail.com');
```

Resultado:

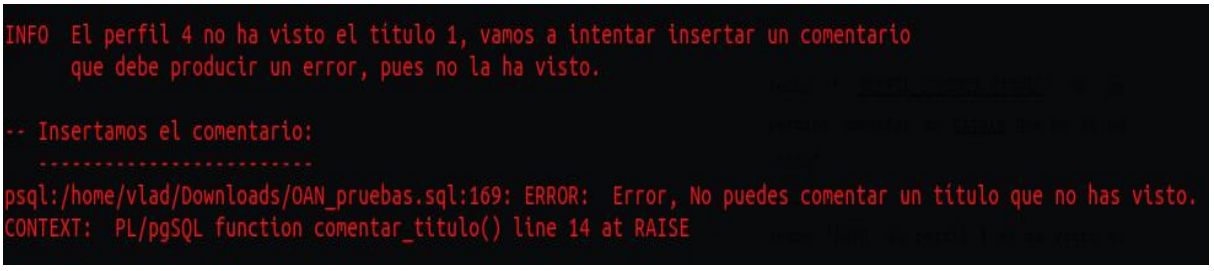
```
INFO  El usuario "julia.almeida@gmail.com" tiene una suscripcion BASICA
      y ya tiene 1 perfil, su máximo es 1.

-- Al intentar insertar otro nos debe dar Error:
   ----------------------------------------
psql:/home/vlad/Downloads/OAN_pruebas.sql:141: ERROR:  Error, Se excede del número máximo de perfiles (1) que permite tu suscripcion (Basica)
CONTEXT:  PL/pgSQL function comprobar_numero_perfiles() line 30 at RAISE
```

Cuarta prueba:
En esta prueba vamos a impedir que un usuario pueda comentar un título que no haya visto.

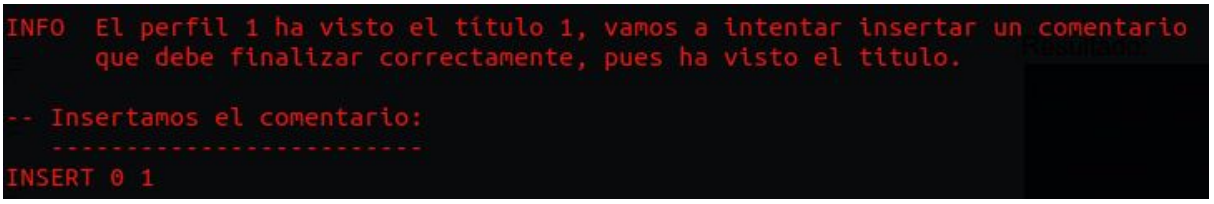
```
 \echo ' PERFIL_COMENTA_TITULO: No se permite comentar un titulo que no se ha
 visto'
 \echo
 \echo 'INFO  El perfil 4 no ha visto el título 1, vamos a intentar insertar
 un comentario'
 \echo '      que debe producir un error, pues no la ha visto.'
```

```
\echo
\echo '-- Insertamos el comentario:'
\echo '   -------------------------'
INSERT INTO perfil_comenta_titulo (idPerfil, idTitulo, fecha, comentario)
VALUES (4, 1, CURRENT_TIMESTAMP, 'No me ha gustado nada la película');
```

Resultado:



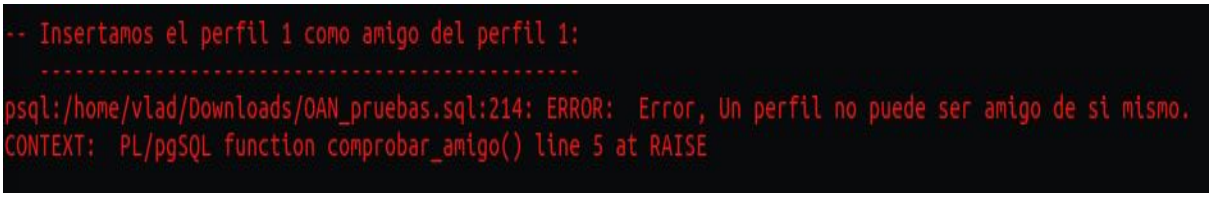
Otro ejemplo:



Quinta prueba:
En esta prueba vamos a impedir que un perfil sea amigo de sí mismo.

```
\echo ' PERFIL_PERFIL: No se permite que un perfil sea amigo de si mismo'
\echo
\echo '-- Insertamos el perfil 1 como amigo del perfil 1:'
\echo '   ------------------------------------------------'
INSERT INTO perfil_perfil (idPerfil, idPerfilAmigo) VALUES (1, 1);
```

Resultado:



Otro ejemplo:

```
-- Insertamos el perfil 1 como amigo del perfil 4 (debe realizarse sin problemas):
--------------------------------------------------------------------------------
INSERT 0 1
================================================================================
```

Sexta prueba:
En esta prueba hacemos que un usuario que haya visto un titulo, y la duracion del titulo visto es aproximadamente igual que la duracion del titulo se marca como visto ese titulo en ese perfil.

```
\echo ' PERFIL_VISUALIZA_TITULO: Si el momento actual coincide
aproximadamente con'
\echo '                          la duración del título, se marca como
visto.     '
\echo
'--------------------------------------------------------------------------
-----'
\echo
\echo '-- Insertamos visualizacion de 1000 segundos de perfil 1 en
titulo 3:'
\echo                                                                    '
----------------------------------------------------------------------'
INSERT INTO perfil_visualiza_titulo (idPerfil, idTitulo,
momento_actual)
VALUES (1, 3, 1000);
\echo
\echo 'RESULTADO:  Debe aparecer como visto = false'
\echo
SELECT * FROM perfil_visualiza_titulo WHERE idPerfil=1 AND idTitulo=3;
```

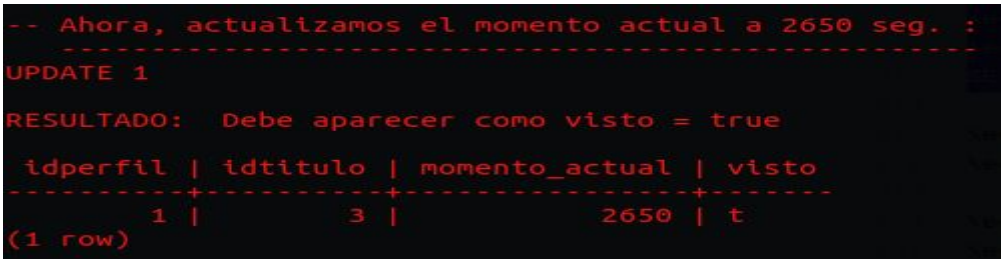
Resultado:

```
-- Insertamos visualizacion de 1000 segundos de perfil 1 en titulo 3:
--------------------------------------------------------------------------------
INSERT 0 1

RESULTADO:  Debe aparecer como visto = false

 idperfil | idtitulo | momento_actual | visto
----------+----------+----------------+-------
        1 |        3 |           1000 | f
(1 row)
```

Y luego modificamos el tiempo:

```
\echo '-- Ahora, actualizamos el momento actual a 2650 seg. :'

UPDATE perfil_visualiza_titulo SET momento_actual = 2650 WHERE idPerfil=1
AND idTitulo=3;
\echo
\echo 'RESULTADO:  Debe aparecer como visto = true'
\echo
SELECT * FROM perfil_visualiza_titulo WHERE idPerfil=1 AND idTitulo=3;
```

Resultado:



Séptima prueba:
En esta prueba comprobamos que si un usuario tiene marcado un título como visto pero lo tiene en descargas este título se elimina.

```
\echo ' -- PERFIL_VISUALIZA_TITULO: Si se marca un título como visto y el
usuario lo tiene en descargas, se elimina.'
\echo
\echo 'INFO: el perfil 1 tiene descargada el titulo 3, pero no lo ha
visto.'
\echo
SELECT * FROM descargas WHERE idPerfil=1 AND idTitulo=3;
\echo
\echo '-- Insertamos visualizacion de 2650 segundos de perfil 1 en titulo
3:'
\echo                                                                    '
----------------------------------------------------------------------'
INSERT INTO perfil_visualiza_titulo (idPerfil, idTitulo, momento_actual)
VALUES (1, 3, 2650);
\echo
\echo 'RESULTADO:  Debe aparecer como visto = true'
\echo
SELECT * FROM perfil_visualiza_titulo WHERE idPerfil=1 AND idTitulo=3;
```

```
\echo
\echo 'RESULTADO:  No debe existir en la tabla descargas'
\echo
SELECT * FROM descargas WHERE idPerfil=1 AND idTitulo=3;
```

Resultado:

El perfil 1 tenia el titulo con id 3 en descargas pero al modificar la tabla perfil_visualiza_titulo añadiendo el tiempo y marcándose como visto se borra de la tabla descargas para ese perfil ese título.

Octava prueba:
En esta prueba tenemos que impedir tenga más de una suscripción activa en el momento actual.

```
\echo ' -- SUSCRIPCION:Impedir que un usuario tenga más de 1 suscripción
activa en el momento actual. '
\echo ' -- SUSCRIPCION:Impedir que un usuario tenga más de 1 suscripción
activa en el momento actual. '
```

```
\echo ' -- SUSCRIPCION:Impedir que un usuario tenga más de 1 suscripción
activa en el momento actual. '
\echo 'INFO: el usuario antonio.gutierrez1984 tiene una solo suscripcion
activa.'
\echo
SELECT * FROM suscripcion WHERE email='antonio.gutierrez1984@gmail.com' and
fecha_finalizacion >= CURRENT_DATE;
\echo
\echo '--Le anadimos al usuario antonio.gutierrez1984 otra suscripcion:'
\echo '-----------------------------------------------------------------'
INSERT INTO suscripcion (idsuscripcion, fecha_finalizacion,
num_descargas_max, tipo_suscripcion, email)
VALUES (5, '2020-05-10', 1, 'Basica', 'antonio.gutierrez1984@gmail.com');
```

Resultado:

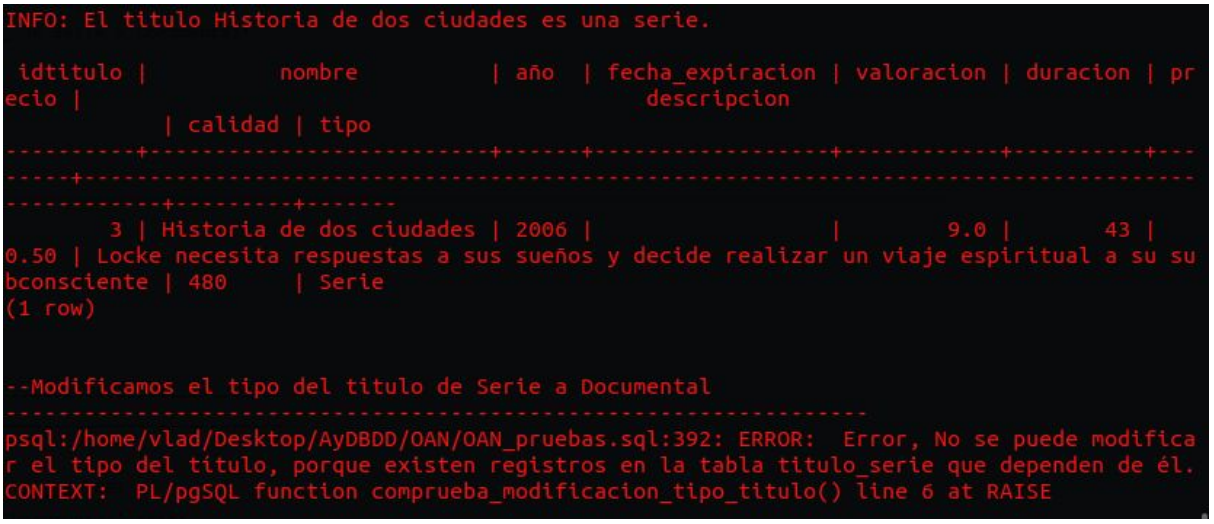
El usuario antonio.gutierrez1984 no puede tener más de una suscripción activa.

Novena prueba:
En esta prueba tenemos que impedir la modificación del tipo de un título, si anteriormente era serie y aún sigue asociada a una serie.

```
\echo ' -- TITULO:Impedir la modificación del tipo de un título, si
anteriormente era serie y aún sigue asociada a una serie'
\echo
\echo 'INFO: El titulo Historia de dos ciudades es una serie.'
\echo
SELECT * FROM titulo WHERE nombre='Historia de dos ciudades' and
tipo='Serie';
\echo
\echo '--Modificamos el tipo del titulo de Serie a Documental'
```

```
\echo
'------------------------------------------------------------------'
UPDATE titulo SET tipo='Documental' WHERE nombre='Historia de dos
ciudades' and tipo='Serie';
```

Resultado:



Décima prueba:
En esta prueba tenemos que comprobar que el título que se desea asociar a una serie es del tipo serie.

```
\echo ' -- TITULO_SERIE:Comprobar que el título que se desea asociar a una
serie es del tipo serie'

\echo 'INFO: El titulo Bad Boys for Life es del tipo Pelicula.'
\echo
SELECT * FROM titulo WHERE nombre='Bad Boys for Life' and tipo='Pelicula';
\echo
\echo '--Introducimos la Pelicula en la tabla TITULO_SERIE'
\echo '------------------------------------------------------------------'
INSERT INTO titulo_serie (idtitulo, capitulo, temporada, idserie)
VALUES (1, 1, 2, 2);
```

Resultado:

```
INFO: El titulo Bad Boys for Life es del tipo Pelicula.

 idtitulo |      nombre       | año  | fecha_expiracion | valoracion | duracion | precio |
    descripcion       | calidad |   tipo
----------+-------------------+------+------------------+------------+----------+--------+-
----------------------+---------+----------
        1 | Bad Boys for Life | 2020 |                  |        6.2 |       90 |   9.99 |
Dos policias muy locos | 1080    | Pelicula
(1 row)


--Introducimos la Pelicula en la tabla TITULO_SERIE
----------------------------------------------------------------------
psql:/home/vlad/Desktop/AyDBDD/OAN/OAN_pruebas.sql:426: ERROR:  Error, El titulo indicado n
o es del tipo Serie.
CONTEXT:  PL/pgSQL function comprueba_serie() line 5 at RAISE
```

Otro ejemplo(cambiando el idserie a otro idtitulo que no sea una Serie):

```
INFO: El titulo Historia de dos ciudades es del tipo Serie y se encuentra en la tabla titul
o_serie.

 idtitulo | capitulo | temporada | idserie
----------+----------+-----------+---------
        3 |        1 |         3 |       1
(1 row)


--Cambiamos a otro titulo
----------------------------------------------------------------------
psql:/home/vlad/Desktop/AyDBDD/OAN/OAN_pruebas.sql:444: ERROR:  Error, El titulo indicado n
o es del tipo Serie.
CONTEXT:  PL/pgSQL function comprueba_serie() line 5 at RAISE
```

Ahora vamos a pasar los checks necesarios:

perfil:

- El número de descargas actuales no puede ser menor que cero.

redsocial:

- El tipo de red social debe ser alguno de los siguientes valores: Facebook, Twitter

suscripción:

- El número de descargas máximas no puede ser menor que cero.
- El tipo de suscripción debe ser alguno de los siguientes valores: Basica, Premium, Contenido

título:

- Valoración debe estar entre los valores 0.0 y 10.0 ambos incluidos.
- La duración del título no puede ser menor que cero.

El primer check comprueba que el número de descargas sea mayor o igual que 0:

```
\echo '-- Actualizamos el perfil 1 con un número de -1 descargas actuales:'
```

```
\echo
\echo 'INFO   Se debe producir un error.'
\echo
UPDATE perfil SET num_descargas_actuales = -1 WHERE idPerfil=1;
```

Resultado:

El segundo check implementado comprueba que el tipo de red social sea o Facebook o Twitter:

```
\echo '-- Actualizamos la redsocial del perfil 3 con tipo Facebook a tipo
Twitter'
\echo 'INFO   Lo debe hace correctamente'
\echo
UPDATE redsocial SET tipo_red_social = 'Twitter' WHERE idPerfil=3;
\echo
\echo 'RESULTADO: '
\echo
SELECT * FROM redsocial WHERE idPerfil=3;
\echo
\echo
\echo '-- Actualizamos la redsocial del perfil 3 con tipo Twitter a tipo Tuenti'
\echo '   ------------------------------------------------------------------------'
\echo
\echo 'INFO   Debe producir un error'
\echo
UPDATE redsocial SET tipo_red_social = 'Tuenti' WHERE idPerfil=3;
```

Resultado:
Si cambias el tipo de red social a uno que no sea Twitter o Tuenti debería saltar el check.

```
-- Actualizamos la redsocial del perfil 3 con tipo Facebook a tipo Twitter
   ----------------------------------------------------------------------

INFO   Lo debe hace correctamente

UPDATE 1

RESULTADO:

   nombre   | tipo_red_social | idperfil
------------+-----------------+----------
 Julia_2000 | Twitter         |        3
(1 row)



-- Actualizamos la redsocial del perfil 3 con tipo Twitter a tipo Tuenti
   ----------------------------------------------------------------------

INFO   Debe producir un error

psql:/home/vlad/Desktop/AyDBDD/OAN/OAN_pruebas.sql:556: ERROR:  new row for relation "redsocial" violates check constraint "redsocial_tipo_red_social_check"
DETAIL:  Failing row contains (Julia_2000, Tuenti, 3).
```

El tercer check comprueba que el número de descargas maximas no sea negativo.

```
\echo ' SUSCRIPCION: El número de descargas máximas no puede ser menor que
cero'
\echo '-- Actualizamos la suscripcion 1 con un número de -1 descargas
máximas:'
\echo                                                                    '
----------------------------------------------------------------------'
\echo
\echo 'INFO   Se debe producir un error.'
\echo
UPDATE suscripcion SET num_descargas_max = -1 WHERE idSuscripcion=1;
```

Si actualizamos el número de descargas maximas a un numero negativo deberia saltar el check.

Resultado:

```
----------------------------------------------------------------------
 SUSCRIPCION: El número de descargas máximas no puede ser menor que cero
----------------------------------------------------------------------


-- Actualizamos la suscripcion 1 con un número de -1 descargas máximas:
   ----------------------------------------------------------------------

INFO   Se debe producir un error.

psql:/home/vlad/Desktop/AyDBDD/OAN/OAN_pruebas.sql:585: ERROR:  new row for relation "suscripcion" violates check constraint "suscripcion_num_descargas_max_check"
DETAIL:  Failing row contains (1, 2020-01-01, -1, Premium, antonio.gutierrez1984@gmail.com).
```

El cuarto check comprueba que el tipo de suscripción sea del tipo Básica,Premium,Contenido y ningún otro tipo:

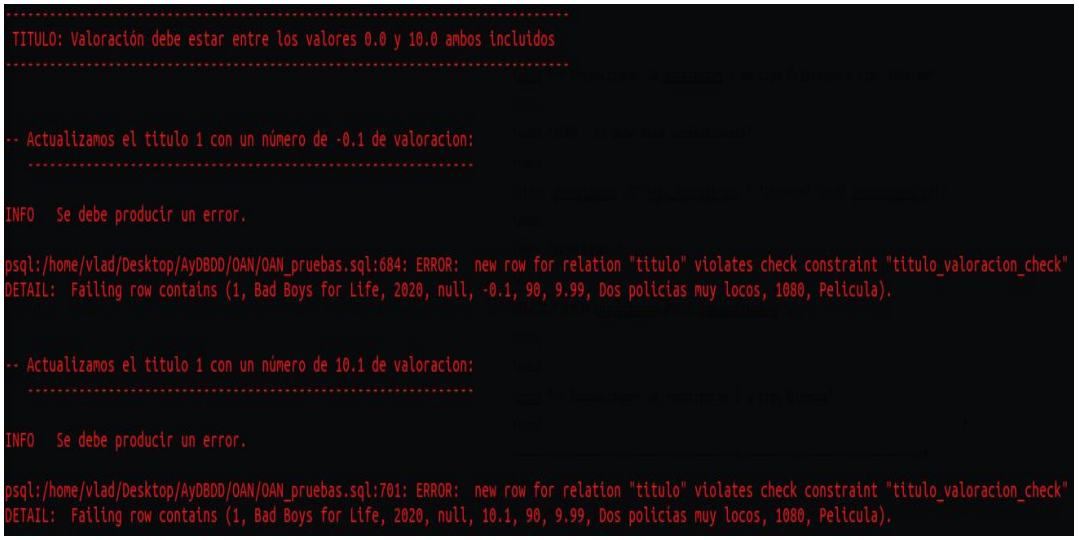
```
\echo '-- Actualizamos la suscipcion 1 de tipo Contenido a tipo Premium'
\echo
\echo 'INFO   Lo debe hace correctamente'
\echo
UPDATE suscripcion SET tipo_suscripcion = 'Premium' WHERE idSuscripcion=1;
\echo
\echo 'RESULTADO: '
\echo
SELECT * FROM suscripcion WHERE idSuscripcion=1;
\echo
\echo
\echo '-- Actualizamos la suscripcion 1 a tipo Ninguna'
\echo                                                                          '
---------------------------------------------------------------------------'
\echo
\echo 'INFO   Debe producir un error'
\echo
UPDATE suscripcion SET tipo_suscripcion = 'Ninguna' WHERE idSuscripcion=1;
```

Resultado:
Al cambiar el tipo de la suscripción a uno que no sea de los tres permitidos salta el check.

El quinto check comprueba que la valoración de un titulo este entre 0.0 y 10.0:

```
\echo '-- Actualizamos el titulo 1 con un número de -0.1 de valoracion:'
\echo
\echo 'INFO    Se debe producir un error.'
\echo
UPDATE titulo SET valoracion = -0.1 WHERE idTitulo=1;


\echo '-- Actualizamos el titulo 1 con un número de 10.1 de valoracion:'


\echo 'INFO    Se debe producir un error.'
\echo
UPDATE titulo SET valoracion = 10.1 WHERE idTitulo=1;
```

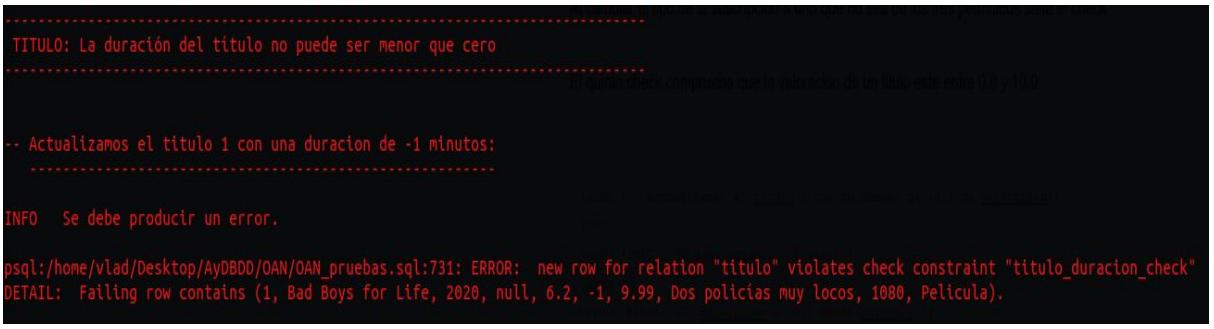
Resultado:



El último check comprueba que la duración del título no puede ser menor que 0:

```
\echo '-- Actualizamos el titulo 1 con una duracion de -1 minutos:'
\echo
\echo 'INFO    Se debe producir un error.'
\echo
UPDATE titulo SET duracion = -1 WHERE idTitulo=1;
```

Resultado:


TITULO: La duración del título no puede ser menor que cero
-- Actualizamos el titulo 1 con una duracion de -1 minutos:
INFO Se debe producir un error.
psql:/home/vlad/Desktop/AyDBDD/OAN/OAN_pruebas.sql:731: ERROR: new row for relation "titulo" violates check constraint "titulo_duracion_check"
DETAIL: Failing row contains (1, Bad Boys for Life, 2020, null, 6.2, -1, 9.99, Dos policías muy locos, 1080, Película).