*

Ejercicio de gestión de usuarios

La gestión de los usuarios en las bases de datos es esencial por varios motivos:

- **Seguridad de los datos**: Permite controlar quién accede a la base de datos y qué acciones puede realizar, protegiendo la integridad y confidencialidad de la información.
- **Cumplimiento Normativo**: Ayuda a cumplir con las estrictas normativas sobre acceso a datos en las organizaciones.
- **Organización y Mantenimiento**: Facilita la identificación de responsabilidades, evita conflictos y permite el seguimiento de actividades y cambios en la base de datos.
- **Facilita la Colaboración**: Permite asignar los roles y permisos adecuados para trabajar en equipo y gestionar los recursos de forma eficiente.
- **Soporte y recuperación**: Al realizar un seguimiento de las actividades de los usuarios, simplifica la corrección de errores y la restauración en caso de problemas en la base de datos.

Tener un sistema adecuado de gestión de usuarios, por tanto, garantiza la seguridad, integridad y eficiencia en la manipulación de datos, facilitando una base de datos segura y colaborativa.

Antes de comenzar, abre MySQL Workbench, conéctate a la base de datos como has hecho en prácticas anteriores, y ejecuta el script SQL que tienes en Moodle (`películas.sql`) para generar la base de datos que se va a utilizar a lo largo de esta sesión práctica.

Gestión de usuarios

Para **crear usuarios** podéis utilizar la sintaxis siguiente:

```
CREATE USER 'nombre_de_usuario' IDENTIFIED BY 'password';
```

Una vez creado, debemos **asignarles los permisos** deseados. Esto lo podemos conseguir con:

```
GRANT PRIVILEGE ON base_de_datos.tabla TO 'nombre_de_usuario'
    [WITH GRANT OPTION];
```

Recordad que:

- `PRIVILEGE` indica los privilegios elegidos de la lista de `CREATE`, `ALTER`, `DROP`, `INSERT`, `UPDATE`, `DELETE` y/o `SELECT`.
- `base_de_datos.tabla` determina el/los schemas y la/las tabla(s) sobre las que se aplican los permisos. Se permite el carácter `*` para aplicar los permisos a más de un schema o tabla. Por ejemplo: `miBD.*` aplica permisos a todas las tablas del esquema `miBD` y `*.*` aplica permisos a todas las tablas de todos los schemas.
- `WITH GRANT OPTION` es opcional, y otorga al usuario la posibilidad de asignatar permisos iguales o inferiores a los suyos a otros usuarios.

Si en algún momento un usuario deja de requerir algún permiso, es importante **revocarlo** mediante la siguiente sentencia:

```
REVOKE PRIVILEGE ON base_de_datos.tabla FROM 'nombre_de_usuario';
```

Además, podemos mostrar los permisos de los que dispone un determinado usuario:

```
SHOW GRANTS FOR 'nombre_de_usuario';
```

Por último, podemos eliminar usuarios:

```
DROP USER 'nombre_de_usuario'
```

Gestión de usuarios mediante código

En esta parte de la práctica vamos a crear un usuario que se llame `usuario_gestor` y le asignaremos permisos de consulta, inserción, modificación y borrado de datos de todas las tablas de nuestra base de datos. Además, le permitiremos otorgar los mismos permisos (o inferiores) a otros usuarios.

Para ello, ejecutaremos las siguientes sentencias:

```
CREATE USER 'usuario_gestor' IDENTIFIED BY 'pass_usr_gstr';
GRANT SELECT, INSERT, UPDATE, DELETE ON peliculas.* TO 'usuario_gestor';
```

Con esto ya tendríamos el usuario `usuario_gestor` creado y con los permisos correspondientes.

Para poder probar los permisos del usuario deberemos crear una nueva conexión en el MySQL Workbench indicando el nombre de usuario del nuevo usuario (Figura 1). Nos pedirá también la contraseña: introducimos lo mismo que introdujimos cuando creamos el usuario.

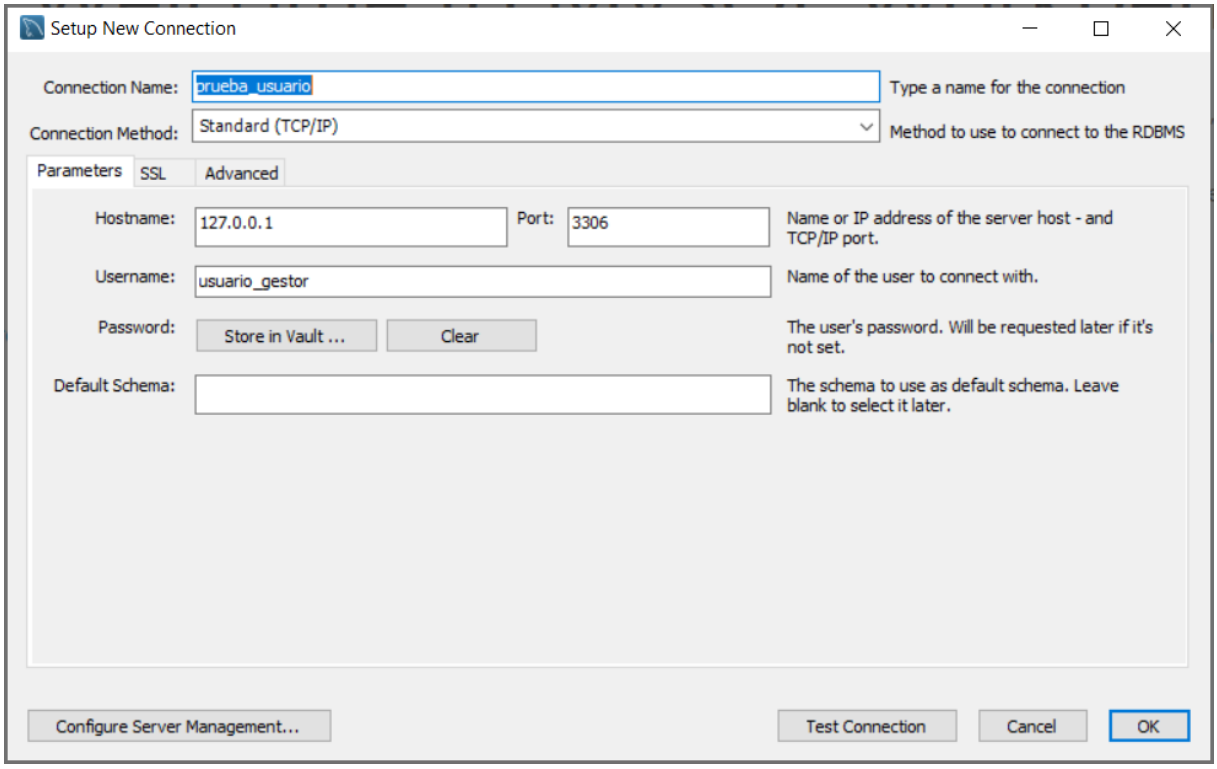

Figura 1: Creación de la nueva conexión para el usuario `usuario_gestor`.

Tras esto, debemos conectarnos al servidor usando la conexión previamente configurada (Figura 2).

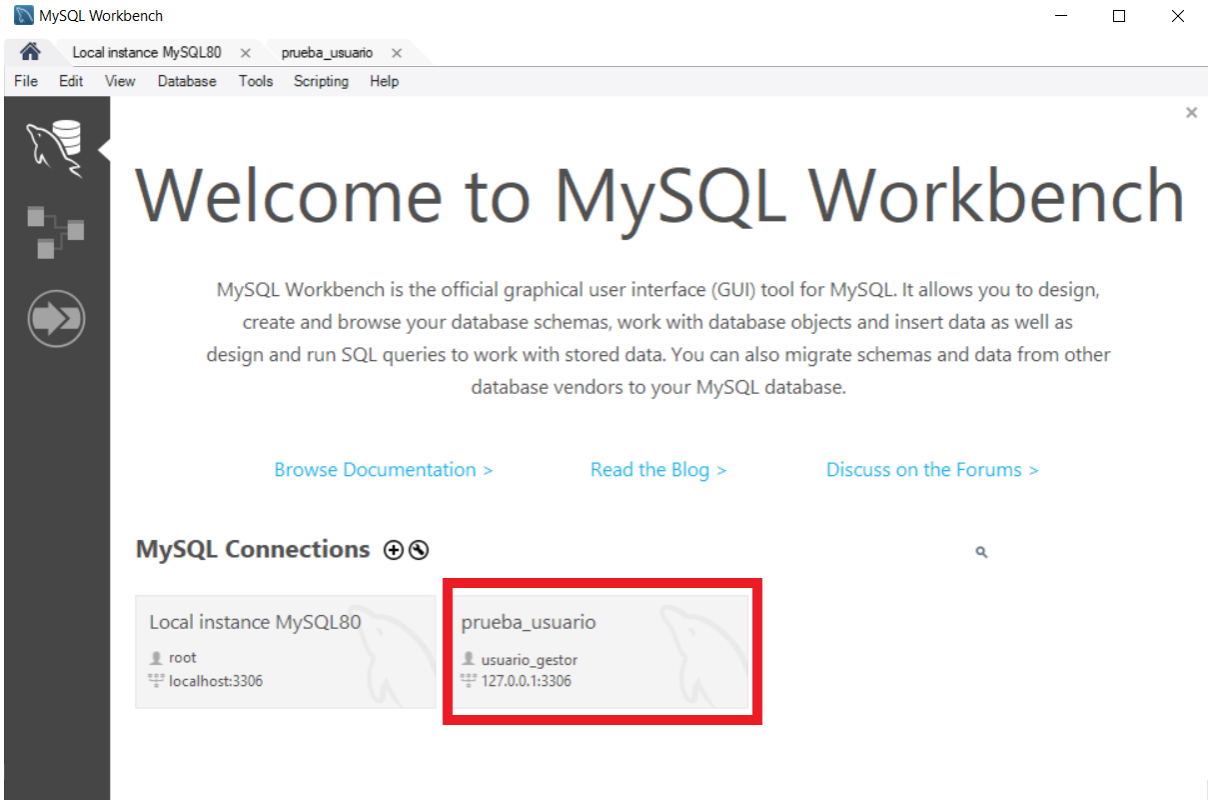

Figura 2: Elección de la conexión con el nuevo usuario.

Si prestamos atención al apartado `schemas`, podremos observar que ahora únicamente está disponible la base de datos `peliculas`, que es a la que le hemos concedido permisos al usuario `usuario_gestor` (Figura 3).

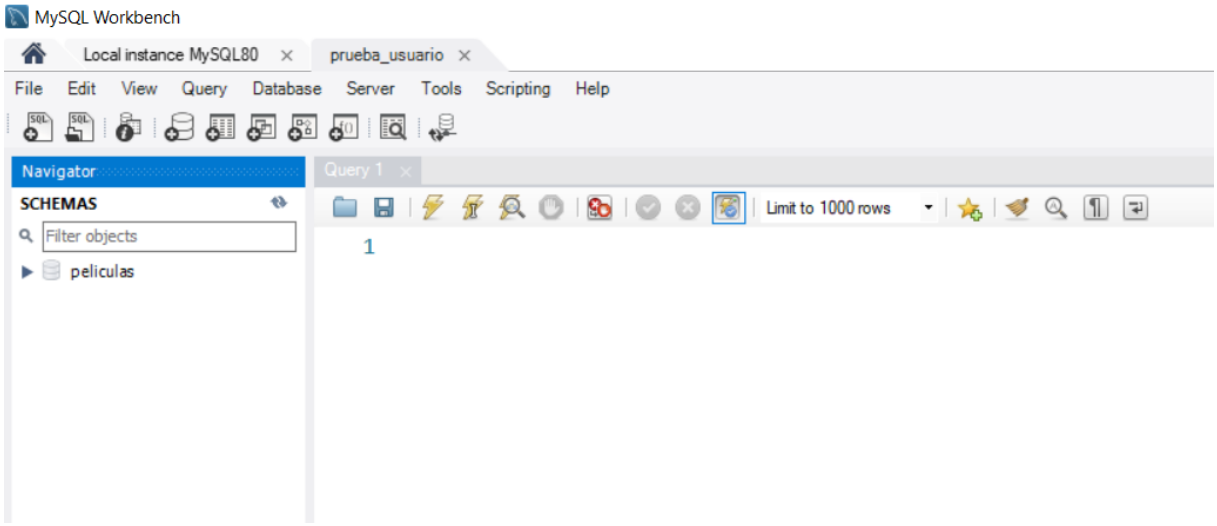

Figura 3: Bases de datos disponibles para el usuario `usuario_gestor`

.

A continuación efectuaremos diferentes operaciones para comprobar que los permisos funcionan correctamente.

En primer lugar, probaremos a ver los registros existentes en la tabla `actor` (Figura 4):

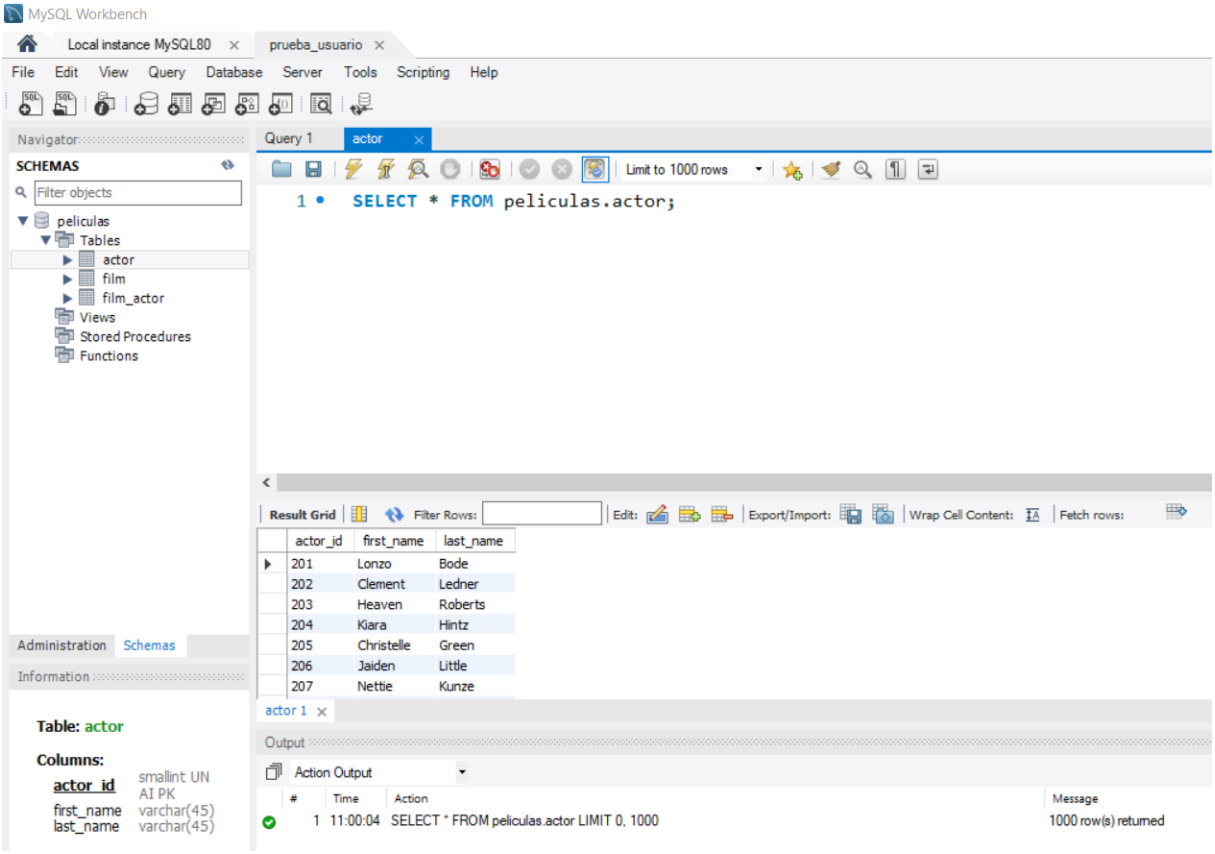

Figura 4: Selección de los registros de la tabla `actor`.

Ahora probaremos a modificar el nombre del actor con `actor_id = 201` (Figura 5):

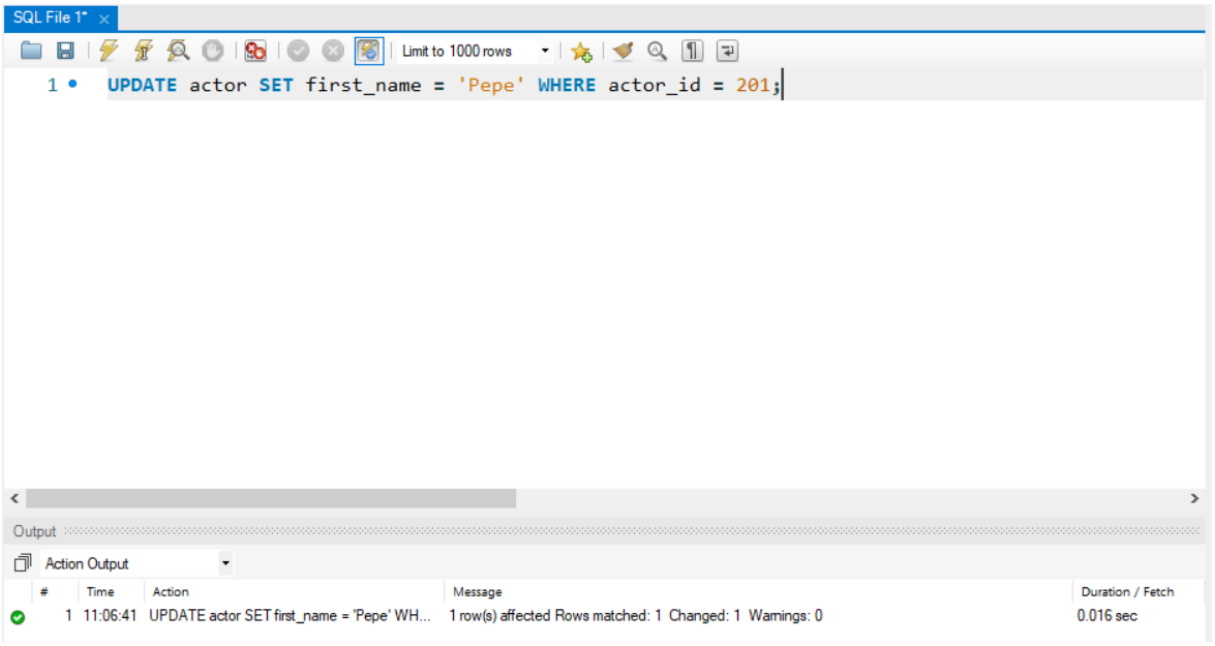

Figura 5: Elección de la conexión con el nuevo usuario.

Como podéis comprobar, éstas operaciones concluyen con éxito, ya que el usuario `usuario_gestor` con el que nos hemos conectado dispone de dichos privilegios (Figura 6)

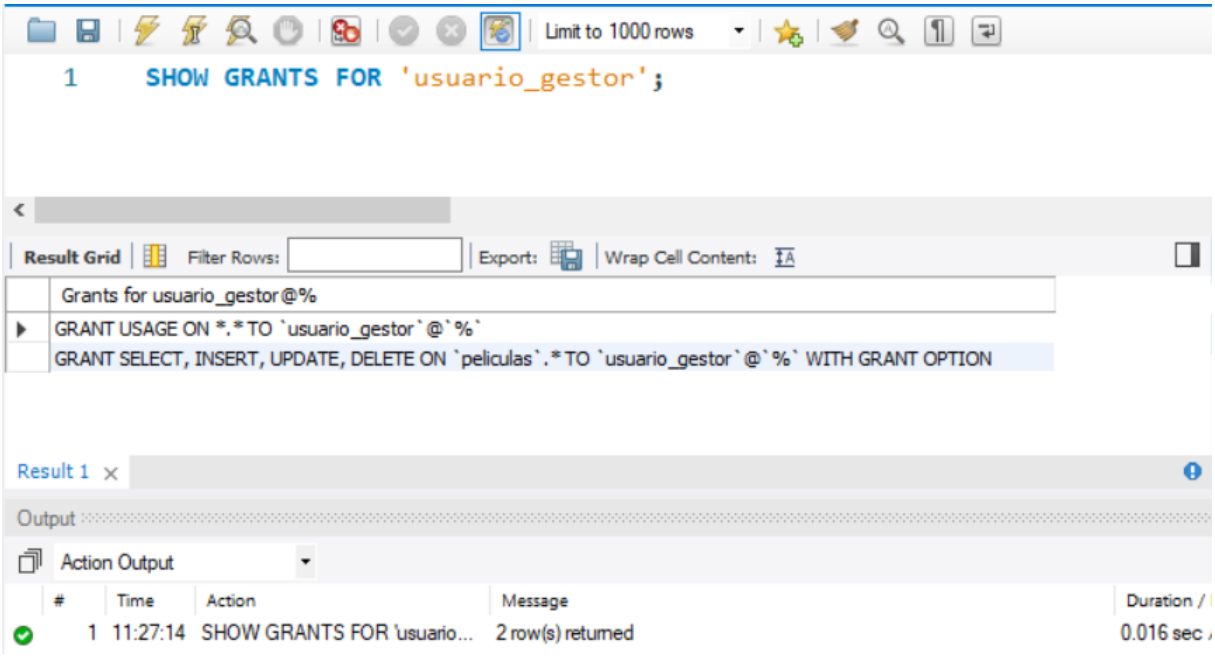

Figura 6: Privilegios del usuario `usuario_gestor`.

Fijaos que la segunda fila indica los privilegios que le hemos otorgado al usuario, pero en la primera fila aparece:

```
GRANT USAGE ON *.* TO 'usuario_gestor' @ '%'
```

Esto no indica que tenga permisos en todos los schemas y tablas, sino todo lo contrario. De acuerdo al manual de MySQL:

> *The USAGE privilege specifier stands for "no privileges." It is used at the global level with GRANT to modify account attributes such as resource limits or SSL characteristics without affecting existing account privileges.*

Es decir, es la forma de MySQL de indicar que existe un usuario, pero no tiene ningún privilegio.

Una vez aclarado esto, vamos a comprobar que efectivamente nuestro usuario no tiene determinados permisos.

Vamos a intentar borrar una tabla:

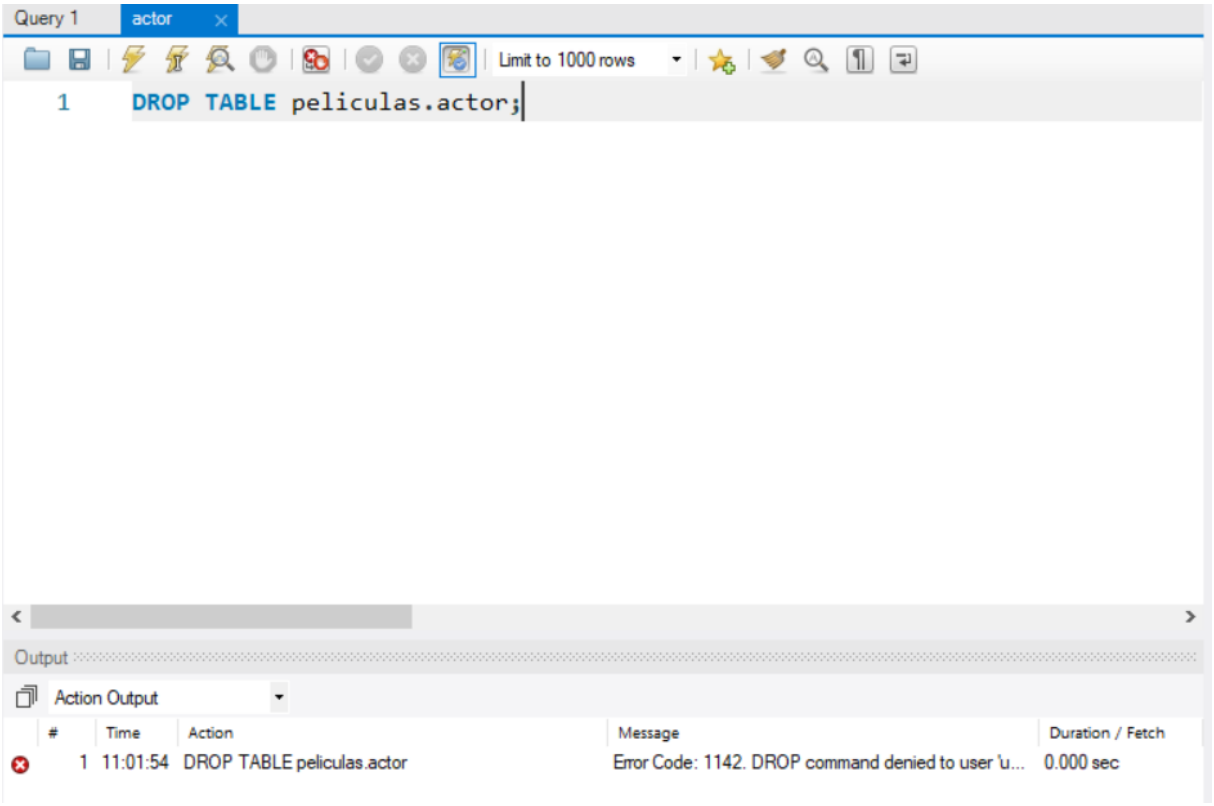

Figura 7: Borrado de una tabla.

Como podéis observar se produce un error debido a que el privilegio `DROP` es denegado al usuario `usuario_gestor`, tal y como debe ser en base a los privilegios que hemos establecido.

Vamos a probar ahora a crear una tabla:

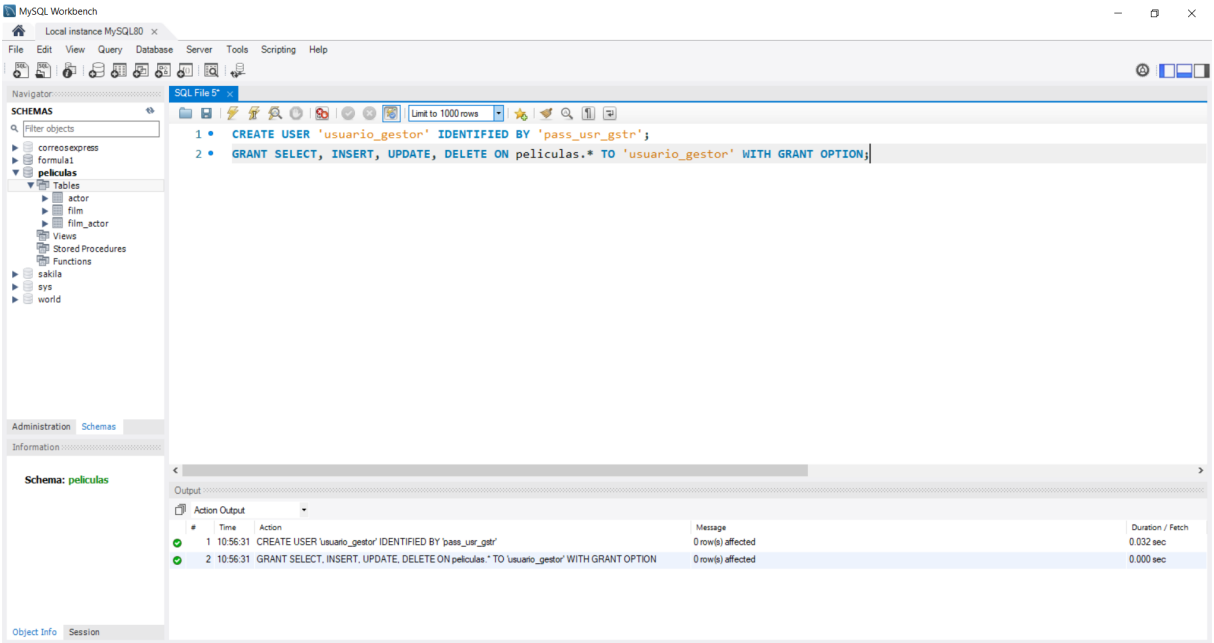

Figura 8: Creación de una tabla.

De nuevo, nuestro usuario no dispone de los permisos necesarios, por lo que no es posible borrarla.

Gestión de usuarios mediante MySQL Workbench

También es posible gestionar los usuarios mediante interfaz gráfica. En esta ocasión, le revocaremos el permiso de modificación a nuestro usuario. Para ello, deberemos ir a *Server* → *Users and privileges*.

Si entráis desde el usuario `usuario_gestor` os encontraréis con que no disponéis de los permisos adecuados (Figura 9):

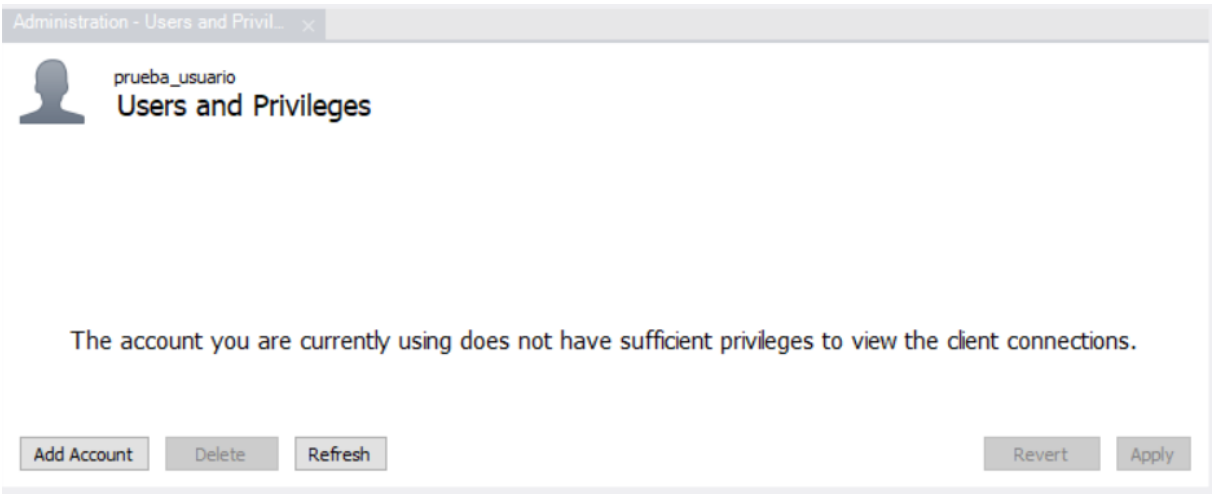

Figura 9: El usuario `usuario_gestor` no dispone de los permisos adecuados para gestionar usuarios.

Por tanto, deberéis utilizar una conexión como `root` para llevar a cabo esta tarea. En ese caso, deberíamos ver algo como esto (Figura 10):

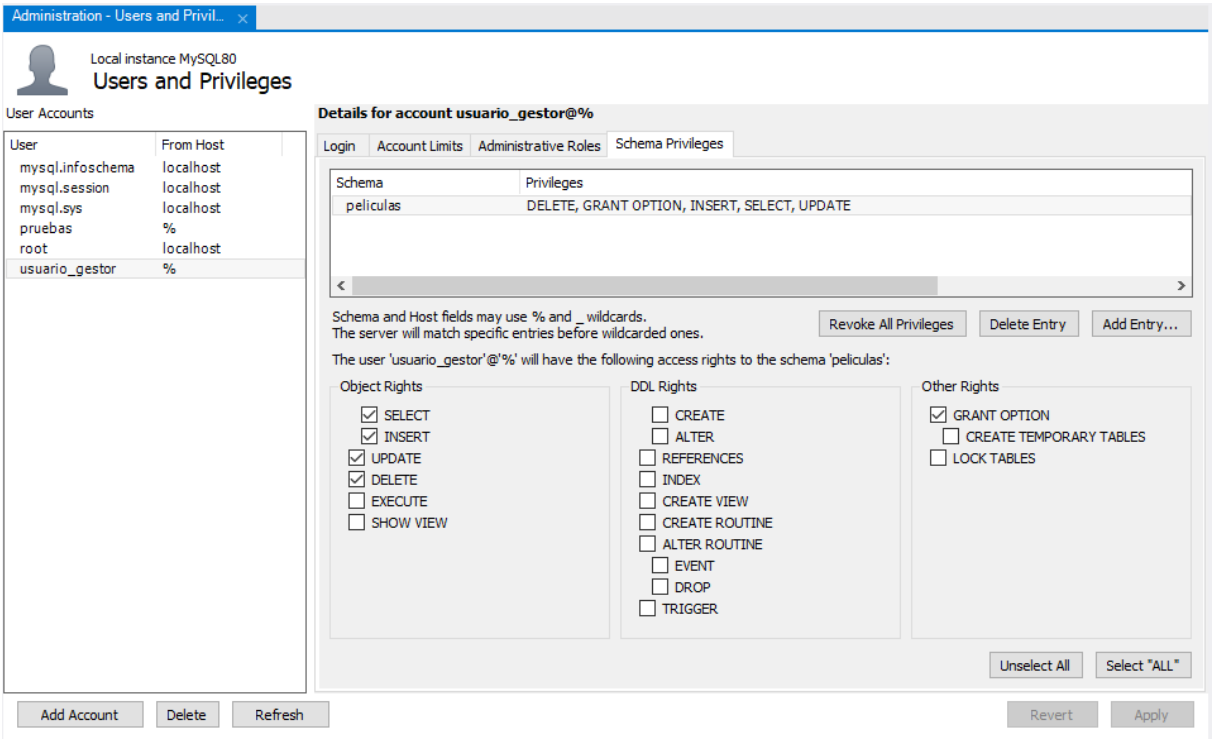

Figura 10: El usuario `root` sí que dispone de los permisos necesarios.

Ahora, revocaremos el permiso de `UPDATE` para el usuario `usuario_gestor` des-seleccionando la opción `UPDATE` y haciendo click en *Apply*.

Si ahora volvemos a conectarnos como `usuario_gestor` y probamos a modificar el nombre del actor con `actor_id = 201` como hicimos antes, comprobaremos que ya no tenemos permiso (Figura 11):

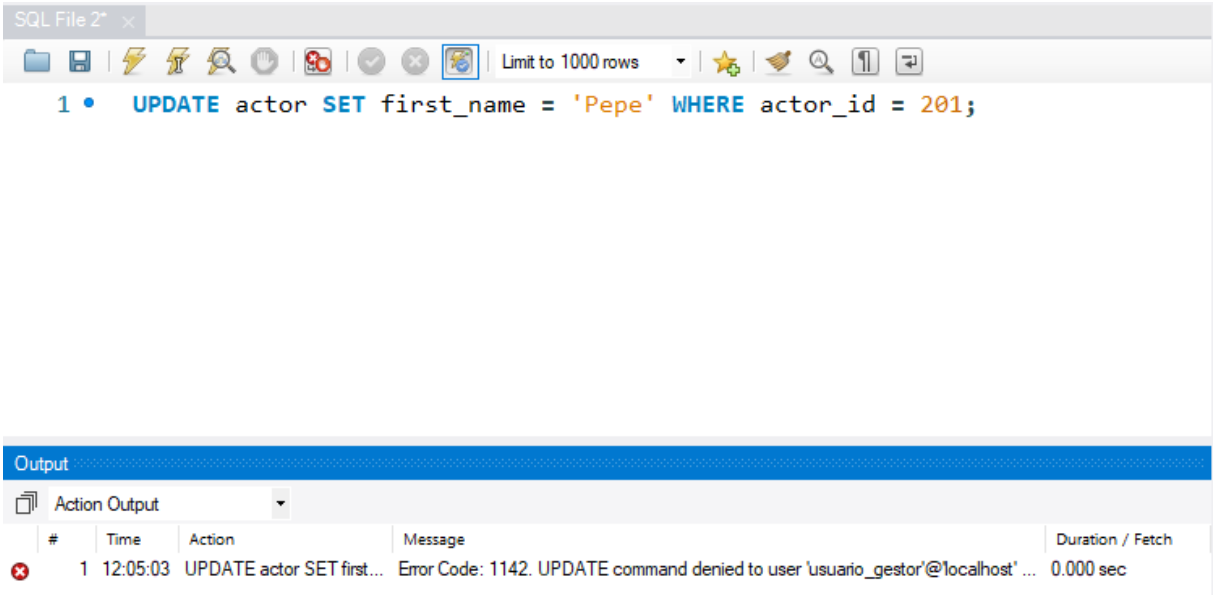

Figura 11: El usuario `usuario_gestor` ya no dispone de permisos para actualizar registros.