Implementación de la base de datos de DataSafe, empresa de seguridad bancaria

Código del proyecto: ICX0_P3

Producto 4 - Usuarios
Resolución de ejercicios

Alumnos: Ana Real Tovar
Jesús Real Tovar
Raúl Vargas Molinero
Marc Sánchez Sanz
Enric Loren Parrondo
Consultor/ra: Rita de la Torre Chirivella
Fecha de entrega: 2 de mayo de 2019

Índice de Contenidos

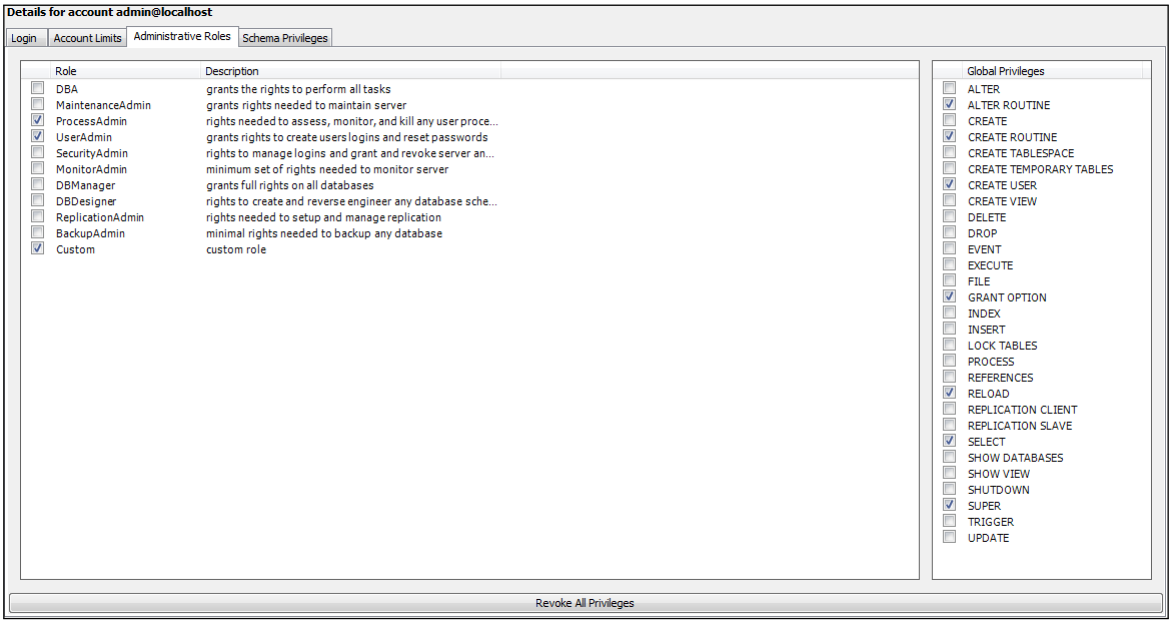
1. Crear el usuario administrador llamado admin, que tendrá todos los privilegios sobre la base de datos de nuestro proyecto.

```
CREATE USER 'admin'@'localhost' IDENTIFIED BY '1234';
GRANT ALL PRIVILEGES ON icxp3_7.* TO 'admin'@'localhost' WITH GRANT OPTION;
GRANT
  CREATE USER,
  RELOAD,
  SELECT,
  CREATE ROUTINE,
  ALTER ROUTINE,
  SUPER
  ON *.* TO 'admin'@'localhost'
  WITH GRANT OPTION;
FLUSH PRIVILEGES;
```

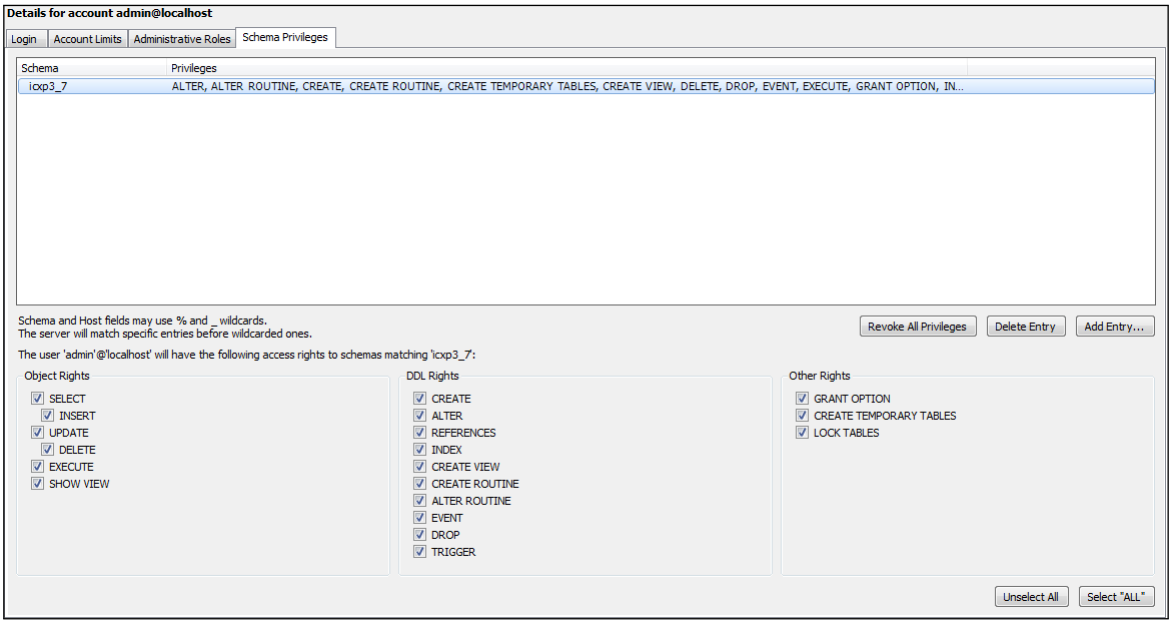
En el primer GRANT, el asterisco después del schema designado donde se otorgan los privilegios, en nuestro caso icxp3_7, se utiliza para definir que el usuario "admin" se crea con total control sobre el schema icxp3_7 (y no otros).

En el segundo GRANT, hacemos que el usuario "admin" pueda crear usuarios de mysql, pueda recargar los permisos(RELOAD), pueda visualizar datos(SELECT) y pueda crear funciones(CREATE ROUTINE, ALTER ROUTINE y SUPER) en toda la instancia de MySQL.

En MySQL Workbench si vamos al menú "Server > Users and Privileges" podemos ver que el usuario "admin" ha sido creado y verificar que tiene todos los privilegios.



Crear el usuario administrador llamado admin, que tendrá todos los privilegios sobre la base de datos de nuestro proyecto.



Otra manera de verificar que se ha creado el usuario “admin” con todos los privilegios es usando las siguientes sentencias:

Para ver que se ha creado el usuario “admin”:

```
1  SELECT user,host FROM mysql.user WHERE user LIKE 'admin';
```

Muestra de la salida:

	user	host
▶	admin	localhost

Para ver sus privilegios:

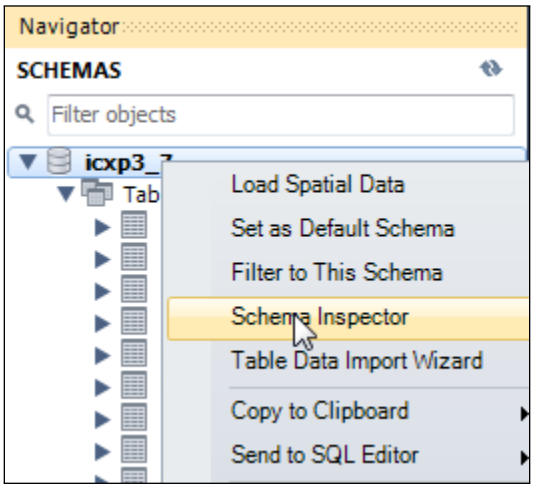
```
1  SHOW GRANTS FOR 'admin'@'localhost';
```

Muestra de la salida:

Grants for admin@localhost
▶ GRANT SELECT, RELOAD, SUPER, CREATE ROUTINE, ALTER ROUTINE, CREATE USER ON *.* TO `admin`@`localhost` WITH GRANT OPTION
GRANT ALL PRIVILEGES ON `icxp3_7`.* TO `admin`@`localhost` WITH GRANT OPTION

Crear el usuario administrador llamado admin, que tendrá todos los privilegios sobre la base de datos de nuestro proyecto.

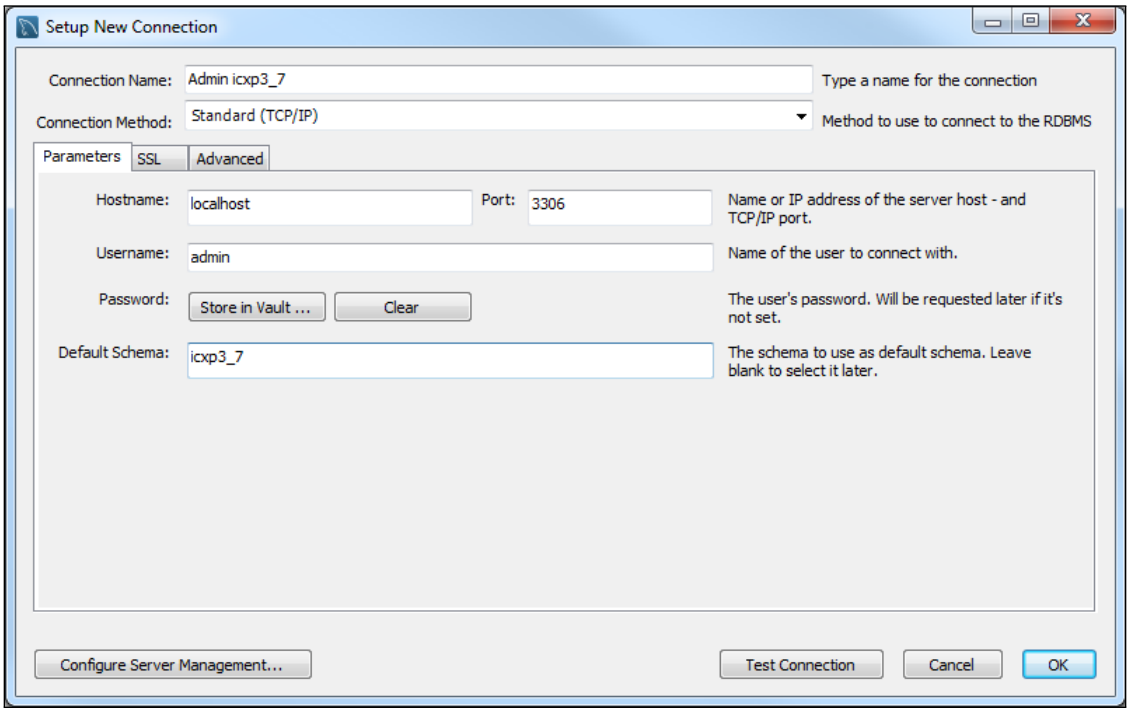
Otra de las formas de verificarlo sería desde la ventana “Navigator” en MySQL Workbench, debemos hacer clic derecho sobre nuestro schema y seleccionar “Schema inspector”.



En el podemos ver información relevante del schema, en la pestaña “Grants” podremos ver los privilegios otorgados al usuario “admin”.

Host	User	Scope	Select	Insert	Update	Delete	Create	Drop	Grant	Refer...	Index	Alter	Creat...	Lock ...	Creat...	Creat...	Alter ...	Execute	Event	Trigger
localhost	admin	<global>	Y	N	N	N	N	N	Y	N	N	N	N	N	N	Y	Y	N	N	N
localhost	admin	icxp3_7	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

2. El usuario administrador crea 2 usuarios más, uno llamado ‘empleado” y el otro llamado ‘agencia’.



Nos conectamos a la base de datos con las credenciales del nuevo usuario “admin” y ejecutamos las siguientes sentencias para crear los usuarios “empleado” y ”agencia”:

```
1    CREATE USER 'empleado'@'localhost' IDENTIFIED BY '1234';
2    CREATE USER 'agencia'@'localhost' IDENTIFIED BY '1234';
```

Podemos verificar que los usuarios se han creado ejecutando la siguiente sentencia:

```
1    SELECT user,host FROM mysql.user WHERE user LIKE 'empleado' OR user LIKE 'agencia';
```

Muestra de la salida:

user	host
agencia	localhost
empleado	localhost

3. A partir de los dos usuarios nuevos creados por el administrador, pensar y dar los privilegios que creas oportunos sobre las tablas y atributos de la base de datos. Justificar la respuesta.

Empleado:

- Actualizar registros en tabla empleado (UPDATE)
- Lectura de registros de cualquier tabla (SELECT)

Para esto ejecutamos las sentencias:

```
1  GRANT UPDATE ON icxp3_7.empleado TO  'empleado'@'localhost';
2  GRANT SELECT ON icxp3_7.* TO  'empleado'@'localhost';
3  FLUSH PRIVILEGES;
```

Podemos verificar que el usuario “empleado” tiene los privilegios con la siguiente sentencia:

```
1  SHOW GRANTS FOR 'empleado'@'localhost';
```

Muestra de la salida:

Grants for empleado@localhost
GRANT USAGE ON *.* TO `empleado`@`localhost`
GRANT SELECT ON `icxp3_7`.* TO `empleado`@`localhost`
GRANT UPDATE ON `icxp3_7`.`empleado` TO `empleado`@`localhost`

Justificación:

En el primer caso, .empleado"podrá ser utilizado por una aplicación, la creación de este usuario se hace para limitar con unos permisos y privilegios concretos las acciones que realizará el usuario de MySQL a partir de las instrucciones que reciba del código de una aplicación. Ya que en MySQL no se puede limitar la consulta o modificación de una sola fila de una tabla, no podemos hacer que un usuario MySQL solo pueda operar con una única fila de una tabla (es decir, no podemos hacer que un único empleado solo pueda ver sus datos desde MySQL, este filtro debe hacerse desde la aplicación).

Creemos que los empleados deben poder actualizar datos relacionados con sus perfiles, como su correo electrónico o su teléfono móvil, para modificarlos en caso de que estos hayan cambiado. También deben de tener permisos de lectura sobre todas las tablas, con tal así de poder desarrollar su operativa laboral de forma eficiente y sin contratiempos, por ejemplo, los trabajadores del departamento de recursos humanos pueden consultar las fechas de contratación de los empleados con las tablas ‘fijo’ y ‘temporal’, los trabajadores del departamento financiero pueden consultar los préstamos solicitados por los empleados desde ‘peticion’ y ‘tipoprestamo’, etc... Aunque en el back-end el usuario ‘empleado’ interactúa con la base de datos, la aplicación realizará un filtrado

a la información que el usuario puede recibir de la base de datos, por ejemplo, los empleados no podrán hacer consultas de tablas que no correspondan a su departamento, los empleados (sin contar excepciones, como el departamento de recursos humanos) sólo podrán consultar información sensible de su propio perfil (en este caso, la aplicación filtra el SELECT del usuario de la base de datos, mostrando solo el registro que coincida con su código de empleado).

Ejemplo:

Todos los empleados de la empresa tendrán una cuenta personal en la aplicación, donde inician sesión con unas credenciales, por ejemplo, dichas credenciales pueden ser su código de empleado y una contraseña que habrán escogido ellos mismos. En el lado del back-end, aunque cada empleado tenga su propia cuenta, todas las cuentas de empleados interactúan con la base de datos utilizando el usuario 'empleado' de MySQL. Finalmente, el filtrado que corresponde según la normativa de protección de datos se realiza desde el lado de la aplicación, que será esta la que debe de limitar que muestre únicamente los datos sensibles que pertenecen al empleado con la sesión iniciada.

Agencia:

- Crear tablas (CREATE)
- Alterar tablas (ALTER)
- Eliminación de tablas (DROP)
- Eliminar registros de cualquier tabla (DELETE)
- Insertar registros en cualquier tabla (INSERT)
- Actualizar registros en cualquier tabla (UPDATE)
- Lectura de registros de cualquier tabla (SELECT)
- Crear vistas (CREATE VIEW)
- Ver las vistas (SHOW VIEW)
- Crear tablas temporales (CREATE TEMPORARY TABLES)

Para esto ejecutamos la sentencia:

```
1  GRANT CREATE, ALTER, DROP, DELETE, INSERT, UPDATE, SELECT, CREATE VIEW,
   ↪ SHOW VIEW, CREATE TEMPORARY TABLES
2    ON icxp3_7.* TO 'agencia'@'localhost';
```

Podemos verificar que el usuario "agencia" tiene los privilegios con la siguiente sentencia:

```
1  SHOW GRANTS FOR 'agencia'@'localhost';
```

Muestra de la salida:

Grants for agencia@localhost
GRANT USAGE ON *.* TO `agencia`@`localhost`
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, CREATE TEMPORARY TABLES, CREATE VIEW, SHOW VIEW ON `icxp3_7`.* TO `agencia`@`localhost`

A partir de los dos usuarios nuevos creados por el administrador, pensar y dar los privilegios que creas oportunos sobre las tablas y atributos de la base de datos. Justificar la respuesta.

Justificación:

En el segundo caso, el usuario “agencia” podrá ser utilizado por una aplicación, la creación de este usuario se hace para limitar con unos permisos y privilegios concretos las acciones que realizará el usuario de MySQL a partir de las instrucciones que reciba del código de una aplicación.

Creemos que las agencias deben poder tener control estructural y poder manipular los datos en el schema, por eso le hemos proporcionado los privilegios necesarios para ello.

Ejemplo:

Las agencias tendrán una cuenta de agencia en la aplicación, donde inician sesión con unas credenciales, por ejemplo, dichas credenciales pueden ser su nombre de agencia y una contraseña elegida por el gerente de la agencia. En el lado del back-end, aunque cada agencia tenga su propia cuenta, todas las cuentas de agencias interactúan con la base de datos utilizando el usuario ‘agencia’ de MySQL. Finalmente, el filtrado que corresponde según la normativa de protección de datos se realiza desde el lado de la aplicación.

4. Realizar una tabla ACL (Access Control Lists) con control de acceso discrecional (DAC), dónde pueda verse la distribución de privilegios para cada usuario.

Rol Usuario	icxp3_7	Admin	Agencia	Empleado
Nivel Control 1: Control total				
Nivel Control 2: Creación de usuarios, funciones y control total en el schema icxp3_7				
Nivel Control 3: Control estructural y manipulación de datos				
Nivel Control 4: Consulta y actualización de datos				