LU09.A01 - SQL- DCL: CREATE User

It is high time to mess a litte around with our new SQL powers, don't you think? So, let's try it directly on our Webstorm.

As the database administrator, we want to create a new user and only grant this user the necessary rights to operate the web application, which includes DML operations such as INSERT, UPDATE, DELETE, but not DDL operations such as CREATE or DROP of tables. After all, we don't want the webapplication take control over our database, are we?

Requirements

- Work type: individual
- Timeframe: 30 Minutes
- Means of aid:
    - only teaching materials, no websearch, no use of ai.
    - Webstorm with connection to the MySQL-DB
- Expected result: Semantically and syntactically correct SQL statements according to the requirements of the case studies.

Case studies / Assignments

As a database administrators we are assigned to create a AppUser, which has for security reasons only the right for DML operations, but must not be allowed for DDL operation. We don't want a hacker to delete our entire webshop, do we?

To get the job done, follow the instructions below:

1. Create as the sysdba (systemadministrator of the databaswe) a new user
2. Grant this role only the necessary rights
3. Create as a sysdba a test table and fill it with some testdata
4. Establish a new connection within the webstorm by using the credentials of this new user
5. Try out the DML operations, wich should be possible (insert, update, delete)
6. Try out to DDL operations, which should result in errors due to missing permissions for that particular user
7. Drop the newly created user finally

Task A1: Prepatory Work

Login as root: Login in from Webstorm to your database as *root* (sysdba = systemadministrator for the datababase) with your a *root-password*.

Task A2: Preparatory Work

New Database: Create a new database named *myDatabase*.

```
CREATE DATABASE myDatabase;
```

Task A3: Preparatory Work

use db: Use that newly created db.

```
USE myDatabase;
```

Task A4: Preparatory Work

Create a test table: Create a table *user* with 3 columns of your choice as a test table, and fill it with some test data.

```
CREATE TABLE user (
  user_id INT AUTO_INCREMENT PRIMARY KEY,
  username VARCHAR(50) NOT NULL,
  email VARCHAR(100) NOT NULL
);

INSERT INTO user (username, email) VALUES ('john_doe',
'john.doe@example.com');
INSERT INTO user (username, email) VALUES ('jane_smith',
'jane.smith@example.com');
INSERT INTO user (username, email) VALUES ('michael_brown',
'michael.brown@example.com');
INSERT INTO user (username, email) VALUES ('sarah_johnson',
'sarah.johnson@example.com');
INSERT INTO user (username, email) VALUES ('chris_williams',
'chris.williams@example.com');
INSERT INTO user (username, email) VALUES ('anna_lee',
'anna.lee@example.com');
INSERT INTO user (username, email) VALUES ('david_kim',
'david.kim@example.com');
INSERT INTO user (username, email) VALUES ('laura_clark',
'laura.clark@example.com');
INSERT INTO user (username, email) VALUES ('jake_lewis',
'jake.lewis@example.com');
INSERT INTO user (username, email) VALUES ('emily_martinez',
'emily.martinez@example.com');
```

Task B1

Create the User: Create a user named *restrictedUser* with the password *SafePassword123* using the *mysql_native_password* plugin.

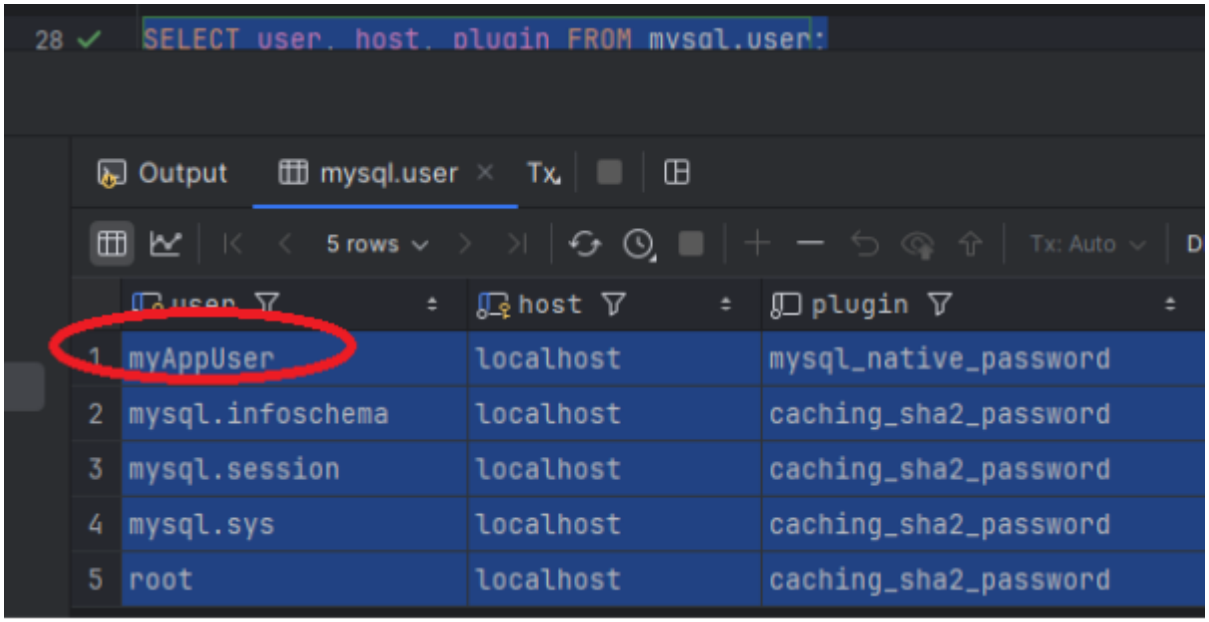
```
CREATE USER 'restrictedUser'@'localhost' IDENTIFIED WITH
mysql_native_password BY 'SafePassword123';
```

Task B2

Overview of the current privileges: Display all users.

```
SELECT user, host, plugin FROM mysql.user;
```

The result set should look like:



Task B2

Grant Privileges Without Table Management: Grant the user *restrictedUser* the ability to read and write data but not to create, alter, or drop tables. Use the following commands to give only the required privileges.

```
GRANT SELECT, INSERT, UPDATE, DELETE ON myAppDB.* TO
'restrictedUser'@'localhost';
```

Task B3

Rovoke Privileges: To be certain that nothing unintended can happen revoke the CREATE, ALTER, and DROP privileges explicitly.

```
REVOKE CREATE, ALTER, DROP ON myAppDB.* FROM 'restrictedUser'@'localhost';
```

Tasc B4

View the User's Privileges: Check the privileges to ensure that the user cannot manage tables.

```
SHOW GRANTS FOR 'restrictedUser'@'localhost';
```

Task B5

Test the User's Access: Connect as *restrictedUser* and try to perform a CREATE TABLE or DROP TABLE operation. The attempt should result in a permission error.

Task B6

Delete the User (optional): After testing, you can delete the user if they are no longer needed.

Solution

Lösung

Vocabulary

English	German
explicitely	ausdrücklich
assignment	Auftrag
to revoke	widerrufen, aufheben