#### **Task 4 - HW: creating and managing roles**

**1. How can one restrict access to certain columns of a database table?**

We can restrict access to certain columns of a database table using column\_privileges.  
Column-level privileges on a table can be granted or revoked using the GRANT and REVOKE commands:  
GRANT {privilege\_type} ON {table\_name}({column\_name}) TO {role\_name}; REVOKE {privilege\_type} ON {table\_name}({column\_name}) FROM {role\_name};  
Managing column-level privileges directly can quickly become complex, especially as the number of users and roles increases but it can be useful in case of accessing sensitive information like credit card numbers or personal identification numbers.  
Column-level privileges can be also achieved via views in PostgreSQL. This approach is often preferred due to its flexibility and ease of use.

**2. What is the difference between user identification and user authentication?**

Identification is the process of recognizing a user, while authentication is the process of verifying the claimed identity of a user.

**Identification** determines the user’s privileges within a system. For example, it specifies which applications can an employee access in a company.

**Authentication** involves the user presenting their credentials for example a password, their biometric data or security token and then the system verifies these credentials against stored records to confirm the user’s identity.

The main difference is that identification is about checking if the user exists, while authentication is about confirming the user’s identity. Identification happens first, and if it passes, then authentication takes place.

**3. What are the recommended authentication protocols for PostgreSQL?**

PostgreSQL supports various authentication protocols, and the choice mainly depends on the security requirements, and the environment:

* Trust Authentication: where PostgreSQL assumes that anyone who can connect to the server is authorized to access the database. Trust authentication is appropriate for local connections on a single-user workstation but it is not secure for production or shared environments.
* Password Authentication include two methods: md5 and password. The password is stored in cleartext in the system. This is a disadvantage because it allows database and system administrators see other users’ passwords. Md5 on the other hand is protected by SSL encryption meaning that the password can be used safely in the database.
* GSSAPI Authentication provides automatic authentication (single sign-on) for systems that support it. The authentication itself is secure, but the data sent over the database connection will be sent unencrypted unless SSL is used.
* SSPI is a Windows technology for secure authentication with single sign-on.
* Kerberos is a network authentication protocol. It provides the tools of authentication and strong cryptography over the network to secure the information systems across an entire enterprise.
* Ident Authentication method works by getting the client's operating system user name from an ident server and then using it to access the database server.
* Peer Authentication is not supported for remote connections as it matches the operating system user with the PostgreSQL user without requiring a password.
* LDAP Authentication works similarly to password authentication except that it uses LDAP as the password verification method. This means, that it is used only to validate the user name/password pairs. Therefore, the user must already exist in the database before LDAP can be used for authentication.
* RADIUS Authentication relies on a RADIUS authentication server and it works by delegating authentication to a server, which handles user credential verification. RADIUS authentication is useful in environments with a large number of users or where centralized policy enforcement is required.
* Certificate Authentication is used in high-security environments and it authenticates users by checking the SSL certificate they send. When using this authentication method, the server will require that the client provide a valid certificate. No password prompt will be sent to the client.
* PAM Authentication operates similarly to password except that it uses PAM (Pluggable Authentication Modules) as the authentication mechanism. PAM is used only to validate user name/password pairs. Therefore the user must already exist in the database before PAM can be used for authentication.
* BSD Authentication delegates user authentication to the BSD authentication subsystem of the operating system. This method relies on external authentication programs provided by the operating system to verify the user’s credentials.

**4. What is proxy authentication in PostgreSQL and what is it for? Why does it make the previously discussed role-based access control easier to implement?**

A database proxy is a database server layer that sits between client applications and the actual database servers, allowing clients to connect directly to the proxy, rather than the database server itself.  
Proxy authentication in PostgreSQL is a mechanism that allows to the user to act on behalf of another user for authentication purposes.

Proxy makes role-based access control easier to implement due to the following reasons:

* traffic can be logged on the proxy server which makes it easier to log and track security incidents
* Users can be authenticated on the proxy server before accessing the database
* Proxys can generate detailed reports on network traffic which can be crucial in identifying and resolving issues
* Proxys enable content filtering, such as blocking unwanted websites or file types.