**Task 4: Answers to Database and Authentication Questions**

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

To restrict access to specific columns in a database table, you can use **role-based access control (**and **column-level permissions**.

* **Grant Permissions Selectively**: Use the GRANT command to give specific roles or users access to only certain columns. For example, in PostgreSQL:

GRANT SELECT (column1, column2) ON table\_name TO role\_name;

This allows role\_name to access only column1 and column2 in table\_name.

* **Revoke Access to Other Columns**: Ensure that the role or user does not have broader permissions that override the column-level restriction. Use REVOKE to remove unwanted access:

REVOKE SELECT ON table\_name FROM role\_name;

GRANT SELECT (column1, column2) ON table\_name TO role\_name;

* **Use Views**: Create a view that exposes only the allowed columns and grant access to the view instead of the table:

CREATE VIEW restricted\_view AS

SELECT column1, column2 FROM table\_name;

GRANT SELECT ON restricted\_view TO role\_name;

* **Row-Level Security** : If you need to restrict rows in addition to columns, enable row-level security (RLS) and define policies.

This approach ensures users only access the columns they are permitted to see, enhancing security.

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

* **User Identification**: This is the process of determining *who* a user claims to be. It answers the question, "Who are you?" Typically, this involves the user providing a unique identifier, such as a username, email, or user ID. For example, entering a username like "luca valentin" identifies the user to the system.
* **User Authentication**: This is the process of verifying that the user is who they claim to be. It answers the question, "Can you prove you are luca valentin Authentication involves providing credentials, such as a password, token, or biometric data, which the system checks against stored records. For example, after entering "luca valentin," the user might provide a password that the system verifies.

**Key Difference**: Identification establishes the user's claimed identity, while authentication confirms the validity of that claim. Identification is the first step, followed by authentication to ensure security.

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

PostgreSQL supports several authentication methods, and the recommended protocols depend on security needs and environment. I recommend this one:

* **SCRAM-SHA-256**: This is the most secure password-based authentication method in PostgreSQL. It uses a challenge-response mechanism with salted passwords, protecting against password sniffing and replay attacks. It’s recommended for modern applications:

# In pg\_hba.conf

host all all 0.0.0.0/0 scram-sha-256

**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?**

* **What is Proxy Authentication in PostgreSQL?** Proxy authentication in PostgreSQL allows a user to connect as one database user (the "proxy") but then act as another user (the target user) without needing to re-authenticate. This is typically done using the SET ROLE command. For example:

*-- User 'proxy\_user' connects, then switches to 'target\_user'*

SET ROLE target\_user;

*That’s all I understand!*