**Part 1.**

Identify and explain the default logins or user accounts for one of the database vendors discussed in the chapter.

A popular database vendor is MySQL. When creating a new MySQL database, it creates a default no password account called “root”. This “root” account has its privileges listed out in the mysql.user grant table. This grant table list outs users accounts and their privileges. So, on the creation of the default “root” user its grant table entry is given explicit allow all privileges.

If you wish to secure your root accounts password, you must create a statement to add a password to it.

An example of this would be.

ALTER USER ‘root’@’localhost’ IDENTIFIED BY ‘thePasswordYouWantForRoot’;

Typically, after the root account is created it will be used to create other user accounts that will be defined with more limiting privileges.

***Part 2.***

**Case Project 6-1: Database Password Policies, page 199**

Create and document a written password policy to be given out to database users in an organization.

**Password Policy**

**Overview:** A strong password policy is vital to an organizations security posture. A poorly chosen password can result in a compromised network. As such, it is expected that all agency employees adhere to these policy guidelines.

**Purpose:** The purpose of this policy is to create a standard for strong passwords and the protection of those passwords along with the data they protect.

**Scope:** The scope of this policy includes all employees from the C-level to the janitors. All members of the organization must follow this policy.

**Policies:**

- All passwords must be changed every 90 days.

- Passwords cannot be the same as any previous 10 passwords.

- Default passwords must be changed on all equipment.

- Passwords should be deleted when a user leaves the company.

- Password should not be written down, stored in a file, and the “remember password” feature should never be used.

- If a password is suspected of compromise report it.

- Passwords should be audited and attempted to crack, if broken the user will be forced to change the password.

- Passwords should never be stored or sent in plaintext.

- Passwords should never be displayed when entered.

**Password Requirements:**

- Should be a minimum of 8 characters.

- Should not be a real word or name.

- Should have at least one capital.

- Should have at least one random character.

- Should have at least one number.

**Penalties:**

Failure to perform this policy can result in a remprimand, termination, and even possible legal action.

**Part 3.**

A database user (or role) is a user (or role) to which the system administrator can assign database privileges. An application user (or role) is a user (or role) created and recognized by an application and cannot be directly assigned database roles or privileges. In multi-tier environments, an authenticated application user must eventually be mapped to a database user or to a database role in order to access database objects. Read "Considerations for Using Application-Based Security" and discuss how a database user can be mapped to application user(s)/role(s). You can describe the consequences on the security of the whole system, its flexibility and performance (using connection pools).

<https://docs.oracle.com/cd/E11882_01/network.112/e36292/app_devs.htm#DBSEG005>

Database users can be mapped to application users and their roles a few ways. One way of doing this is called “One Big Application User” model. In this model applications user authenticate themselves to the application and then the application connects itself as a highly privileged user. This makes permission relatively easy but can be detrimental in the auditing process because database transactions are logged as a single user instead of the application user they logged in as, another issue is that an attacker could bypass the application altogether and make statements directly in the database without communicating through the application. Besides from the One Big Application User model, database roles can be tied to the application roles by assigning permissions to the users directly. This allows for a more individualized permissioning but can become quite cumbersome quickly. The more users the more permissioning must be done. One benefit is the audit trail will have information about the individual users. Lastly there is a mixture of both were permissioning can be grouped out. Certain application users can belong to a group with specific permissions. This can be more manageable but less specific when it comes to permissioning. Also, the audit trail will only tell you the group and not the individual user which may or not be helpful depending on the audit.