UniSENAI

Módulo 7: Segurança e Controle de Acesso

Módulo 7: Segurança e Controle de Acesso

A segurança de dados é uma das prioridades em sistemas de banco de dados. O MySQL oferece ferramentas robustas para gerenciar usuários e suas permissões, garantindo que os dados sejam acessados apenas por quem possui autorização. Este módulo aborda como criar, gerenciar e proteger contas de usuários no MySQL, além de implementar boas práticas de segurança.

---

1. Gerenciamento de Usuários

O gerenciamento de usuários no MySQL é essencial para garantir que apenas pessoas ou sistemas autorizados acessem o banco de dados.

Criação de Usuários (`CREATE USER`)

O comando `CREATE USER` cria uma nova conta de usuário no MySQL, especificando o nome do usuário e a senha.

Sintaxe Geral

```
CREATE USER 'nome_usuario'@'host' IDENTIFIED BY 'senha';
```

- **`nome_usuario`**: Nome do usuário.
- **`host`**: Define de onde o usuário pode se conectar (`localhost` ou `%` para qualquer lugar).
- **`senha`**: Define a senha do usuário.

Exemplo Prático

Criar um usuário chamado `usuario1` com acesso local:

```
CREATE USER 'usuario1'@'localhost' IDENTIFIED BY 'senha_forte123';
```

Criar um usuário chamado `usuario_remoto` com acesso remoto:

```
CREATE USER 'usuario_remoto'@'%' IDENTIFIED BY 'senha_remota123';
```

Alteração e Exclusão de Usuários

Alteração de Usuários (`ALTER USER`)

O comando `ALTER USER` é usado para modificar atributos do usuário, como a senha.

Exemplo Prático: Alterar a senha de `usuario1`:

```
ALTER USER 'usuario1'@'localhost' IDENTIFIED BY 'nova_senha_forte456';
```

Exclusão de Usuários (`DROP USER`)

O comando `DROP USER` remove permanentemente uma conta de usuário do MySQL.

Exemplo Prático: Excluir o usuário `usuario_remoto`:

```
DROP USER 'usuario_remoto'@'%';
```

2. Concessão e Revogação de Permissões

Os comandos `GRANT` e `REVOKE` são usados para conceder ou revogar permissões aos usuários, definindo o que eles podem ou não fazer no banco de dados.

Comando `GRANT`

O comando `GRANT` concede permissões específicas a um usuário.

Sintaxe Geral

```
GRANT tipo_de_permissao ON nome_do_banco.nome_da_tabela TO 'nome_usuario'@'host'
```

- **`tipo_de_permissao`**: Permissões como `SELECT`, `INSERT`, `UPDATE`, `DELETE`, etc.
- **`nome_do_banco.nome_da_tabela`**: Define o escopo das permissões.

Exemplo Prático

Conceder permissão de leitura (`SELECT`) na tabela `clientes` para `usuario1`:

```
GRANT SELECT ON meu_banco.clientes TO 'usuario1'@'localhost';
```

Conceder todas as permissões em um banco de dados:

```
GRANT ALL PRIVILEGES ON meu_banco.* TO 'usuario1'@'localhost';
```

Comando `REVOKE`

O comando `REVOKE` remove permissões previamente concedidas.

Sintaxe Geral

```
REVOKE tipo_de_permissao ON nome_do_banco.nome_da_tabela FROM 'nome_usuario'@'host';
```

Exemplo Prático

Revogar a permissão de leitura na tabela `clientes` de `usuario1`:

```
REVOKE SELECT ON meu_banco.clientes FROM 'usuario1'@'localhost';
```

Permissões de Acesso

Permissões comuns em MySQL incluem:

- **`SELECT`**: Permite consultar dados.
- **`INSERT`**: Permite inserir novos registros.
- **`UPDATE`**: Permite modificar registros existentes.
- **`DELETE`**: Permite remover registros.

---

3. Melhores Práticas de Segurança

Manter um banco de dados seguro envolve mais do que gerenciar usuários e permissões. É crucial adotar práticas de segurança robustas.

Uso de Senhas Fortes e Políticas de Senhas

- **Senhas Fortes**:
  - Devem conter pelo menos 8 caracteres.
  - Devem incluir letras maiúsculas, minúsculas, números e símbolos.
  - Exemplo: `Senha$F0rte!2024`
- **Configurar Políticas de Senhas**: Ative a política de senha forte no MySQL para garantir senhas seguras:

```
SET GLOBAL validate_password_policy = 'STRONG';
```

Configuração de Permissões de Usuário de Forma Segura

1. **Princípio do Menor Privilégio**:
   - Conceda ao usuário apenas as permissões necessárias.
   - **Exemplo**: Um usuário que só precisa ler dados deve receber apenas permissão `SELECT`.

```
GRANT SELECT ON meu_banco.* TO 'leitor'@'localhost';
```

2. Restringir Acessos:

- Especifique o `host` de onde o usuário pode acessar.
- **Exemplo**: Permitir acesso apenas do servidor local:

```
CREATE USER 'usuario_local'@'localhost' IDENTIFIED BY 'senha123';
```

3. Evitar Usuários Compartilhados:

- Cada pessoa ou sistema deve ter sua própria conta de usuário.

4. Auditoria de Usuários e Permissões:

- Regularmente revise as contas e permissões para identificar excessos.

Aplicabilidade Prática

1. **Sistemas de Gestão de Vendas**:
    - Um usuário pode ter acesso apenas às tabelas relacionadas aos pedidos, enquanto outro pode acessar apenas os clientes.

```
CREATE USER 'leitor_prod'@'%' IDENTIFIED BY 'senhaLeitor!';
GRANT SELECT ON producao.* TO 'leitor_prod'@'%';
```

2. Ambientes Multiusuário:

- Desenvolvedores podem receber permissões diferentes para realizar testes em tabelas específicas, enquanto administradores têm acesso total.

3. Ambientes de Produção:

- Use usuários separados para leitura e escrita para evitar alterações acidentais em dados críticos.


```
CREATE USER 'leitor_prod'@'%' IDENTIFIED BY 'senhaLeitor!';
GRANT SELECT ON producao.* TO 'leitor_prod'@'%';
```


Ao implementar estas práticas, é possível garantir que o banco de dados esteja protegido contra acessos não autorizados, mantendo a integridade e confidencialidade das informações.

Vamos Praticar

Questões Práticas

1. **Criação de Usuários**:
   Crie dois usuários no MySQL:
    - Um usuário chamado `usuario_local` com acesso apenas local (`localhost`).
    - Um usuário chamado `usuario_remoto` com acesso remoto (`%`).
2. **Alteração de Usuários**:
   Modifique a senha do usuário `usuario_local` para uma senha forte, seguindo as práticas recomendadas.
3. **Exclusão de Usuários**:
   Exclua o usuário `usuario_remoto` do banco de dados.
4. **Concessão de Permissões**:
    - Conceda permissão de leitura (`SELECT`) na tabela `clientes` para o usuário `usuario_local`.

    - Conceda todas as permissões em um banco de dados chamado `loja` para o usuário `admin_user`.
5. **Revogação de Permissões**:
   Revogue a permissão de leitura (`SELECT`) na tabela `clientes` do usuário `usuario_local`.
6. **Configuração de Permissões por Host**:
   Crie um usuário chamado `usuario_servidor` com permissão de acessar o banco de dados apenas do servidor local.
7. **Princípio do Menor Privilégio**:
   Configure um usuário chamado `relatorio_user` que tenha acesso somente à tabela `relatorios` com permissões de leitura (`SELECT`).
8. **Auditoria de Usuários**:
   Liste todos os usuários criados no banco de dados e suas permissões utilizando os comandos adequados.
9. **Políticas de Senhas**:
   Ative a política de senhas fortes no MySQL e aplique-a ao usuário `usuario_local`.
10. **Configuração em Ambientes de Produção**:
    Configure dois usuários em um banco de dados de produção:
    - Um usuário com permissões apenas de leitura (`SELECT`).
    - Um usuário com permissões completas para inserção e alteração (`INSERT`, `UPDATE` e `DELETE`).

Questões Teóricas

1. Explique a diferença entre os comandos `GRANT` e `REVOKE` no MySQL.
    - Dê exemplos de situações em que cada comando seria usado.
2. O que é o princípio do menor privilégio? Por que ele é importante em sistemas de banco de dados?
3. Qual é a importância de usar senhas fortes em bancos de dados? Cite as características de uma senha segura.
4. Por que é recomendável evitar o uso de usuários compartilhados em bancos de dados?
5. Quais são os riscos de conceder permissões desnecessárias a um usuário? Explique como evitar esses riscos.
6. Explique como a restrição de acesso por host pode aumentar a segurança do banco de dados.
7. Descreva as permissões comuns em MySQL (`SELECT`, `INSERT`, `UPDATE`, `DELETE`) e dê exemplos de suas aplicações.
8. O que é uma auditoria de usuários e permissões? Por que é importante realizá-la regularmente em um banco de dados?
9. Quais práticas devem ser seguidas ao conceder permissões em um ambiente multiusuário?

10. Considere um ambiente de produção onde há uma equipe de desenvolvedores e uma equipe de administradores. Como você configura os usuários e permissões para garantir segurança e eficiência?

Instruções para a Entrega das Atividades

1. **Elaboração e Envio do Arquivo**
   - Responda todas as questões de forma clara e objetiva.
   - Gere um arquivo no formato **.PDF** contendo as respostas de cada questão.
   - Envie o arquivo para os e-mails dos professores responsáveis.
2. **Validação da Atividade**
   - Após o envio do arquivo, procure o(s) professor(es) para realizar a validação da atividade.
   - **Não inicie a próxima atividade sem antes validar a anterior com o professor.**
3. **Forma de Validação**
   - **Explicação Verbal**: Explique cada resposta verbalmente ao(s) professor(es).
   - **Perguntas e Respostas**: Esteja preparado para responder aos questionamentos do(s) professor(es) sobre o conteúdo das respostas.
   - **Orientação**: Receba orientações sobre a apresentação do(s) tema(s).