

Banco de dados II

Aula 3

Setembro de 2024

Segurança em Bancos de dados

- Introdução à segurança
- Usuários e papéis
- Papeis ROLES
- Permissões de acesso





01

>>>> Introdução de segurança

Banco de dados

Introdução segurança em banco de dados

Uma das maiores preocupações em computação tem sido segurança da informação

Nos dias atuais, com o uso da Internet os sistemas tornam-se onipresentes, entretanto também vulneráveis a ataques maliciosos

Portanto, os SGBDs trazem uma camada de segurança que visa compor toda o arsenal de segurança da informação numa corporação

Introdução segurança em banco de dados

Definição:

- Segurança em Banco de dados diz respeito à proteção do banco de dados contra ataques intencionais ou não intencionais utilizando-se ou não de meios computacionais

Áreas envolvidas:

- roubo e fraude
- perda de confidencialidade e privacidade
- perda de integridade
- perda de disponibilidade

Introdução segurança em banco de dados

O subsistema de segurança é responsável por proteger o BD contra o acesso não autorizado.

Formas de acesso não autorizado:

- leitura não autorizada
- modificação não autorizada
- destruição não autorizada

Introdução segurança em banco de dados

O DBA tem plenos poderes para dar e revogar privilégios a usuários.

- Criação de contas
- Concessão de privilégios
- Revogação de privilégios
- Atribuição do nível de segurança

Introdução segurança em banco de dados

Motivação: Exemplo Atacado

- Apenas alguns empregados podem modificar preços dos produtos
- Clientes usando o sistema de consulta, não devem ter acesso a outras funcionalidades (vendas, contabilidade, folha de pagamento, etc)
- Apenas o pessoal da gerência deve ter acesso às informações dos empregados (por exemplo: empregados-a-demitir)
- Clientes não devem ver o preço de compra de um produto

Introdução segurança em banco de dados

Controles de segurança computacionais

- Adiciona-se uma camada à segurança provida pelo SO
- Autorização e autenticação
- Views
- Backup e recovery

Introdução segurança em banco de dados

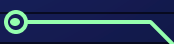
- Integridade
- Stored procedures
- Criptografia
- Auditoria
- Procedimentos associados
 - e.g. upgrading, virus checking, proxy, firewall, kerberos, certificados digitais, SSL, SHTTP, etc.



Introdução segurança em banco de dados



Controles de segurança não computacionais

- Política de segurança e plano de contingência
 - Posicionamento seguro de equipamentos
 - Controle de acesso físico
 - Manutenção
- 

Introdução segurança em banco de dados

Duas abordagens para segurança de dados:

- Controle de acesso discreto:
 - Um dado usuário tem direitos de acessos diferentes (privilégios) em objetos diferentes
 - Flexível, mas limitado a quais direitos usuários podem ter em um objeto
- Controle de acesso mandatório:
 - cada dado é rotulado com um certo nível de classificação
 - A cada usuário é dado um certo nível de acesso
 - rígido, hierárquico

Em SQL 1999 temos:

Proteção	Privilégio	Aplica-se a
Ver	SELECT	Tabelas Métodos invocados
Criar	INSERT	Tabelas
Modificar	UPDATE	Tabelas, colunas
Remover	DELETE	Tabelas
Referenciar	REFERENCES	Tabelas, colunas
Usar	USAGE	UDT(User-Defined Type)
Ativar	TRIGGER	Tabelas
Executar	EXECUTE	Stored procedures

Introdução segurança em banco de dados

O que se espera do SGBD é o mesmo tratamento dada à tentativa de acesso a uma tabela inexistente (“no such table”).

Portanto, se um usuário tentar acessar uma tabela que ele não tem privilégios para tal o erro será:

“Either no such table or you have no privilege on the table”

Razão: Segurança

Introdução segurança em banco de dados

O usuário tem um auth_ID que o identifica

Existe PUBLIC que representa todos usuários

Privilégios são atribuídos/revogados:

- Usuários
- Papéis (Roles)

O criador de um objeto é o dono do objeto e assim tem todos os privilégios sobre o objeto, podendo autorizar a outros usuários alguns(ou todos) destes privilégios.

A opção with grant option, permite ao usuário que recebeu um privilégio repassar para quem quiser.

Introdução segurança em banco de dados

Com respeito a DDL:

- Um usuário pode executar qualquer comando DDL no seu esquema que ele possui.
- Um usuário NÃO pode executar nenhuma operação DDL no esquema que ele não possui



02

Usuários e papéis



Usuários e papéis

Identificador de usuário

- Alguns SGBDs permitem que o usuário use o mesmo login e senha do SO

Papéis (Roles)

- É um identificador ao qual pode-se atribuir privilégios que não existem a princípio. Então pode-se atribuir a um usuário este papel (conjunto de privilégios) com um único comando GRANT.
- Pode-se inclusive ao criar um papel usar outros papéis já cadastrados.
- Ex. PapelVendedor, PapelVendedorSapatos, PapelVendedorFrutas.

Usuários e papéis

Exemplo de pilha de autorizações:

AuthID	Role name	
-	-	
José	Null	Stored procedure
Null	Vendedor	SQL
Carlos	Null	Login

03

Papeis ROLES

Usuários - USER

Sintaxe SQL:1999

CREATE USER 'nome-papel'@'servidor' IDENTIFY BY 'senha';

Para remover um papel:

DROP USER nome-papel;

Papeis - roles

Existem papéis padrões na maioria dos SGBD:

- DBA: permite desempenhar o papel de administrados do banco de dados
- Resource: permite criar seus próprios objetos
- Connect: permite apenas se conectar ao banco de dados, mas deve receber os privilégios de alguém para acessar objetos.

Regras de autorização

Expressam os mecanismos de autorização em relações/visões/ stored procedures

São compiladas e armazenadas no dicionário de dados

São expressas em linguagem de alto nível (Ex. SQL)

Uma maneira do SGBD implementar estas regras é usar uma matriz de autorização, onde cada linha corresponde a um usuário e cada coluna corresponde a um objeto.

$M[i,j] \Rightarrow$ conjunto de regras de autorização que se aplica ao usuário i com relação ao objeto j .

Regras de autorização

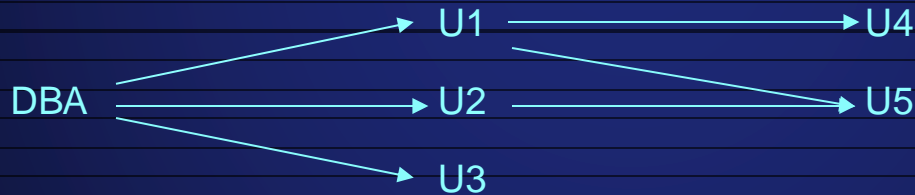
Exemplo:

Indivíduo	Empregado	Departamento	Projeto
João	SELECT	SELECT, UPDATE	SELECT, DELETE, UPDATE
Bianca	NONE	NONE	SELECT
Pedro	NONE	NONE	NONE
Ana	ALL	ALL	ALL

O ABD fornece/revoga as autorizações de leitura, inserção, atualização e remoção aos usuários nas diversas tabelas/visões, e estes podem repassá-los caso receba autorização para tal.

Autorização em Banco de dados

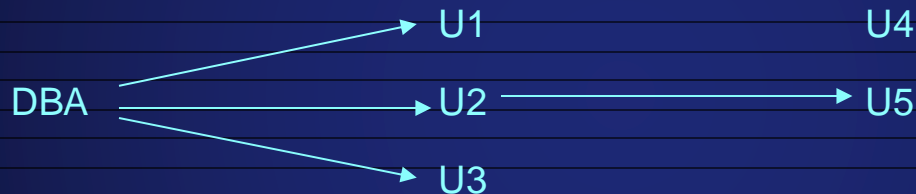
Um usuário que tem concedida alguma forma de autoridade pode passar essa autoridade para outros usuários.



OBS.: Para um usuário criar outro usuário atribua a ele CREATE USER. Segue o exemplo a seguir:

Autorização em Banco de dados

Suponha que o administrador do banco de dados decida revogar a autorização do usuário U1.

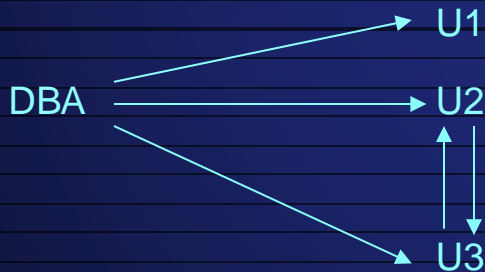


Uma vez que o usuário U4 tem a autorização concedida pelo usuário U1, a sua autorização também será revogada.

No entanto, U5 mantém sua autorização por ela ter sido concedida também por U2.

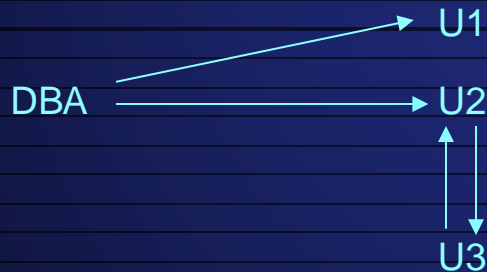
Autorização em Banco de dados

Um par de usuário desonestos pode tentar burlar as regras anteriores de revogação de autorização concedendo autorização de um para outro.



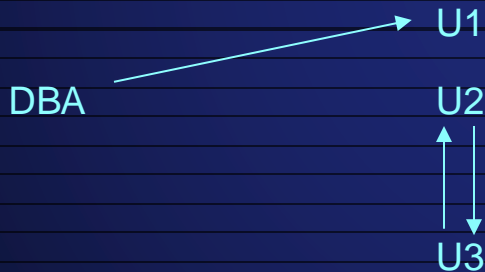
Autorização em Banco de dados

Se o administrador do banco de dados revogar a autorização de U2, este mantém sua autorização através de U3.



Autorização em Banco de dados

Se a autorização é revogada subsequentemente de U3, U3 retém sua autorização através de U2.



Autorização em Banco de dados

Para evitar problemas como esse, requeremos que todas as arestas num grafo de autorização sejam parte de algum caminho originado no administrador do banco de dados.





04

>>>> Permissões de acesso em SQL

Permissão de Acesso em SQL

O Comando **GRANT** é usado para conferir autorização. A forma básica deste comando é:

grant <lista de privilégios>

on <nome da relação ou visão>

to <lista de usuários>

IMPORTANTE:

FLUSH PRIVILEGES;

Permissões de acesso em SQL

Os privilégios a serem autorizados, em SQL, com o comando grant são:

- **SELECT** → Autorização leitura
 - Na linguagem SQL original era usado o privilégio read
- **INSERT** → Autorização inserção
- **UPDATE** → Autorização atualização
- **DELETE** → Autorização eliminação

Permissões de acesso em SQL

INDEX → Autorização índice

REFERENCES → Autorização recursos

ALTER → Autorização alteração

DROP → Autorização remoção

Exemplo de permissões

GRANT SELECT ON agencia TO U1, U2, U3

GRANT UPDATE ON deposito TO U1

GRANT REFERENCES (nome-agencia) ON agencia TO U1

Criação de Novos usuários

A forma correta de se atribuir o privilégio de criação de novos usuários é:

```
GRANT CREATE USER ON *.* TO usuario1;
```

E para remover:

```
REVOKE CREATE USER ON *.* FROM usuario1;
```

Permissão de Acesso em SQL

Em SQL, um usuário a quem é garantido um privilégio não está autorizado a conceder aquele privilégio a outro usuário. A fim de conceder um privilégio, e permitir ao receptor passar o privilégio a outros usuários, a cláusula `with grant option` é acrescentada ao comando `grant` apropriado.

```
GRANT SELECT ON agencia TO U1 WITH GRANT OPTION;
```

Criação de uma Role

CREATE ROLE [nome];

Dando privilégios:

GRANT SELECT, INSERT, UPDATE, DELETE ON [base.tabela] TO [nome da role];

Adicionando um usuário a uma role:

ALTER USER [nome do usuario] ADD MEMBER [nome da role];

Atribuindo um usuário a uma Role

```
CREATE ROLE [nome];
```

```
GRANT [nome] TO [usuario];
```

Associação de usuário a Role

GRANT role TO [user,] [role,]

Exemplo

Grant role_x TO user_x;

Permissão de Acesso em SQL

Para revogar a autorização, o comando revoke é usado. Ele toma a forma quase idêntica àquela do comando grant:

```
revoke <lista de privilégios>  
on <nome da relação ou visão>  
from <lista de usuários>
```

Exemplos de Acesso em SQL

revoke select on agencia from U1, U2, U3

revoke update on deposito from U1

revoke references (nome-agencia) on agencia from U1

Bancos de dados estáticos

Um banco de dados que permite queries que derivam informação agregadas(e.g. somas, médias)

- Mas não queries que derivam informação individual

Tracking

- É possível fazer inferências de queries legais para deduzir respostas ilegais
- e.g. WITH (STATS WHERE SEXO='M' AND FUNCAO = 'Programador') AS X : COUNT(X)
- WITH (STATS WHERE SEXO = 'M' AND FUNCAO = 'Programador' AS X : SUM(X,SALARIO)

Visões SQL

Nesta aula introduziremos o conceito de visão em SQL, mostraremos como são especificadas, discutiremos o problema das atualizações e como ela pode ser implementada em um SGBD.

Conceito de visão em SQL

Uma visão, na terminologia SQL, é uma tabela única e derivada de outra tabela, que pode ser uma tabela básica ou uma visão previamente definida. Uma visão não existe de forma física, ela é considerada uma tabela virtual, em contraste com as tabelas básicas, cujas tuplas são realmente armazenadas no banco de dados. Isso limita as operações de atualização possíveis para as visões, embora não imponha nenhuma limitação para consultas.

Especificando visões em SQL

Em SQL, o comando para especificar uma visão é o CREATE VIEW. A visão recebe o nome (virtual) de tabela (ou nome da visão), uma lista de nomes de atributos e uma consulta para especificar o conteúdo dessa visão. Se nenhum dos atributos da visão for resultado de uma função ou de uma operação aritmética, não teremos de especificar os nomes de atributos para a visão, dessa forma, eles receberiam como padrão os mesmos nomes que os atributos das tabelas de definição.

Especificando visões em SQL

Uma view nada mais é do que instruções SELECT já pré-definidas e armazenadas no banco.

Este conceito pode parecer simples, e é, mas pode-se resolver muitos problemas com as views.

Uma view não é um programa, portanto dentro dela você só pode escrever o comando Select.

Especificando visões em SQL

Você pode colocar dentro de uma view o comando select feito com os seguintes critérios:

- Consultas com união (Join)
- Consultas com união (Union)
- Consultas com Group By com função de totalização
- Consultas com subconsultas (Subquery)
- Consultas com Order By apenas com Top.

Exemplo de VIEW

Para criar uma view, utilizamos o comando CREATE VIEW.

```
CREATE VIEW <view_name>
```

```
AS
```

```
<instrução_SELECT>;
```

Exemplo de VIEW

Como exemplo vamos criar uma view que apresente o pedido e o nome do cliente.

```
CREATE view v_pedido as
```

```
Select numerodopedido, nomedaempresa
```

```
    datadopedido
```

```
From  clientes a, pedidos b
```

```
Where a.codigodocliente = b.codigodocliente;
```

Como fica o comando select após a criação da View.

```
Select * from v_pedido;
```

Especificando visões em SQL

Supõe-se que uma visão esteja sempre atualizada; se modificarmos as tuplas das tabelas básicas sobre as quais a visão foi construída, a visão deverá, automaticamente, refletir essas alterações. Consequentemente, a visão não é realizada no instante de sua definição, mas quando especificarmos uma consulta sobre ela. É responsabilidade do SGBD, e não do usuário, ter a certeza de que uma visão está atualizada.

Excluindo uma Visão

Se não precisarmos mais de uma visão podemos usar o comando SQL “DROP VIEW” para dispensá-la.

Por exemplo, para nos livrar da visão, podemos usar a seguinte declaração SQL:

```
DROP VIEW TRABALHA_EM1;
```

Implementação e atualização Visão

As implementações eficientes de visões para as consultas são um problema complexo. Existem duas abordagens principais. Uma estratégia, chamada modificação da consulta, implica modificar uma consulta de visão em uma consulta de tabelas básicas. A desvantagem dessa abordagem é que ela é ineficiente para as visões definidas via consultas complexas que tenham a execução demorada, precisamente no caso de serem aplicadas diversas consultas à visão dentro de um curto espaço de tempo.

Outra estratégia, chamada materialização da visão, implica criar fisicamente uma tabela temporária a partir da primeira consulta a essa visão e mantê-la, considerando que poderão seguir-se outras consultas. Nesse caso, deve-se ter uma estratégia eficiente para a atualização da tabela da visão sempre que as tabelas básicas forem atualizadas, de modo a garantir que a visão esteja sempre atualizada.

05

Permissões de acesso

Atualização e Visão

Uma visão pode esconder dados que o usuário não necessita, ou não deve, ver.

Esta propriedade serve tanto para simplificar o uso do sistema como para aumentar a segurança.

- O uso do sistema é simplificado porque o usuário pode restringir sua atenção aos dados de interesse.
- A segurança é fornecida se existir um mecanismo para restringir o usuário a uma ou mais visões pessoais.

Atualização e Visão

Os sistemas de banco de dados relacionais tipicamente fornecem segurança em dois níveis:

- Relação: um usuário pode ter acesso a uma relação permitido ou negado.
- Visão: um usuário pode ter acesso aos dados que aparecem em uma visão permitido ou negado



.....

Obrigado!

Alguma dúvida?
jheymesso.Cavalcanti@unicap.br