13

Segurança de bancos de dados e transações

No capítulo 12, verificamos como criar usuários no banco de dados e estabelecer um critério de segurança de acesso às informações. Uma vez criado o usuário, ele não possui qualquer direito sobre o banco de dados. Para que se possa acrescentar e revogar direitos aos usuários, devemos utilizar comandos específicos.

Segurança é um item primordial que diferencia banco de dados SQL de outros bancos de dados. Por meio de critérios rígidos de controle, podemos ter uma informação segura e que apenas usuários autorizados podem alterá-la.

Embora os direitos variem de um banco de dados para outro, os comandos relacionados à atribuição e à revogação de direitos são os mesmos. Vamos nos concentrar, portanto, nos comandos e não no tipo de direito.

Normalmente os bancos de dados vêm com um ou dois usuários (SYS, SYSTEM, DBA ou SA são exemplos desses usuários) que têm poder sobre todo o banco de dados. Portanto, devemos utilizar esses usuários para criar outros usuários e estes, por sua vez, receberão direitos de sistema e concederão direitos de objeto a outros usuários.

Classificação dos direitos

Existem duas classificações para os direitos no banco de dados:

1. Objeto: diz respeito aos objetos que um *schema* possui.
2. Sistema: diz respeito ao que se pode realizar no banco de dados.

Direitos de objeto

Como é o *schema* que possui os objetos, somente ele poderá conceder os direitos sobre seus objetos a outros usuários do banco de dados. Assim, os demais usuários somente poderão realizar o que lhes for permitido pelo proprietaário do objeto.

Privilégio de Objeto	Tabela	Visão
ALTER	SIM	
DELETE	SIM	SIM
INSERT	SIM	SIM
REFERENCES	SIM	
RENAME	SIM	SIM
SELECT	SIM	SIM
UPDATE	SIM	SIM

Tipos de direitos

Existem seis tipos básicos de direitos de acesso (podendo haver mais ou menos, dependendo do banco de dados):

1. SELECT: permite extrair dados das tabelas ou visões.
2. INSERT: permite incluir dados em tabelas ou visões. Alguns bancos de dados permitem que apenas algumas colunas possam ser atualizadas.
3. UPDATE: permite modificar linhas nas tabelas. Alguns bancos de dados permitem que apenas algumas colunas possam ser atualizadas.
4. DELETE: permite excluir linhas de tabelas ou visões.
5. REFERENCES: permite criar CONSTRAINT de chave estrangeira.
6. ALL PRIVILEGES: concede todos os privilégios anteriores.

Atribuindo direitos de objeto

Para atribuir direitos a outros usuários, utilizamos o comando GRANT.

```
GRANT tipo_direito
   ON tabela
   TO {usuário|PUBLIC}
   [WITH GRANT OPTION];
```

Quando utilizamos a cláusula WITH GRANT OPTION, significa que será transferida ao usuário que recebe o direito a possibilidade de ele dar os mesmos direitos que recebeu a outros usuários. Exemplo:

```
GRANT SELECT
    ON CD
    TO SCOTT;

GRANT SELECT
    ON CD
    TO SCOTT WITH GRANT OPTION;
```

Caso desejemos atribuir direitos de pesquisa em toda tabela e alteração somente na coluna NOME_CD, utilizaremos o seguinte comando:

```
GRANT SELECT, UPDATE (NOME_CD)
    ON CD
    TO SCOTT;
```

Caso queiramos atribuir direitos a todos os usuários do banco de dados, utilizamos PUBLIC em vez do nome do usuário:

```
GRANT SELECT
    ON CD
    TO PUBLIC;
```

Revogando direitos de objeto

Para revogar direitos anteriormente concedidos, utilizamos o comando REVOKE:

```
REVOKE tipo_direito
   ON tabela
   FROM usuário;
```

Para revogar o direito de pesquisa na tabela CD, utilizamos o comando:

```
REVOKE SELECT
    ON CD
    FROM SCOTT;
```

Direitos de sistema

Estes direitos também variam muito em função do banco de dados utilizado. Contudo, os mais comuns são:

a. CONNECT: permite ao usuário conectar-se ao banco de dados. Usuários que não tem esse direito sequer podem acessar o banco de dados.

b. RESOURCE: permite ao usuário, além de conectar-se, criar objetos no banco de dados.

c. DBA: permite ao usuário controlar todos os objetos do banco de dados. É um direito que deve ser concedido com muito critério para evitar problemas ao banco de dados. Recomenda-se que apenas o administrador do banco de dados o possua.

Privilégio de Sistema	Usuário	Índice	Procedimento	Tabela	Visão
ALTER	SIM	SIM	SIM	SIM	
CREATE	SIM	SIM	SIM	SIM	SIM
DROP	SIM	SIM	SIM	SIM	SIM
SELECT				SIM	
EXECUTE			SIM		
UPDATE				SIM	

Atribuindo direitos de sistema

Utilizamos o mesmo comando GRANT:

```
GRANT privilégio
   TO usuário
   [WITH ADMIN OPTION];
```

Exemplo:

```
GRANT CONNECT TO SCOTT;
```

Ao utilizarmos a cláusula WITH ADMIN OPTION, estamos dando direito ao usuário de transferir o mesmo direito recebido a outros usuários.

Revogando direitos de sistema

Utilizamos o mesmo comando REVOKE:

```
REVOKE privilégio
   FROM usuário;
```

Sessões de banco de dados

Quando um usuário interage com o banco de dados, ele precisa estar conectado a ele. O tempo compreendido entre a entrada e a saída do banco de dados é denominado sessão do banco de dados.

Para conectar-se ao banco de dados, utilizamos o comando:

```
CONNECT
```

Como cada banco de dados possui sua própria ferramenta de acesso, quando não forem pedidos usuário e senha em uma janela-padrão, utilizar o comando CONNECT deve ser o suficiente. Normalmente, são solicitados usuário e senha após esse comando.

Para desconectar-se do banco de dados, utilizamos o comando:

```
DISCONNECT
```

Transações e níveis de isolamento

Uma transação é uma unidade de trabalho submetida ao banco de dados. É comum que mais de um usuário realize transações concorrentes, ou seja, ao mesmo tempo. Teoricamente, seria possível determinar transações que fossem realizadas em *série* (cada transação completa o trabalho antes que a outra inicie) ou *inter-relacionada* (em que as ações de todas as transações são alternadas).

O resultado de operações inter-relacionadas pode não ocasionar nenhum problema e concluir-se como em uma operação em série. Mas imagine que duas pessoas realizem operações iguais (consulta de estoque de material, por exemplo) em frações de segundo diferentes. Corre-se o risco de a última das consultas retornar a quantidade errada em função da atualização do primeiro usuário. Para isso, deve haver um controle de bloqueio de linhas.

Contudo, esse bloqueio pode afetar o desempenho do banco de dados, visto que o segundo usuário seria deixado em *wait state* (estado de espera) até que a primeira transação fosse concluída. Esse tipo de bloqueio é conhecido por *exclusive lock*. Há ainda bloqueios do tipo *shared* e *read*. O padrão é *shared*, para permitir o máximo de operações concorrentes sem que haja bloqueio das operações.

Dependendo do banco de dados, pode haver bloqueios de linha, páginas e tabelas. Quanto menor o nível do bloqueio (no caso, linha) mais operações concorrentes podem ocorrer sem haver perda de disponibilidade do banco de dados. Há bancos de dados que prometem bloqueio de colunas em futuro próximo. Entretanto, isso naturalmente faz com que o gerenciador do banco de dados tenha que trabalhar mais e, conseqüentemente, consumir mais tempo para realizar a tarefa.

Foi visto anteriormente neste livro que uma transação no banco de dados pode ser encerrada de duas formas: gravando ou restaurando a situação anterior. O comando para gravar é o COMMIT e para restaurar é o ROLLBACK. Deve-se considerar que o ROLLBACK recuperará todas as transações efetuadas desde o último COMMIT. Logo, é conveniente que constantemente sejam gravadas as informações no banco de dados, pois não há como selecionar quais operações não devem ser gravadas.

SAVEPOINT e ROLLBACK TO

Este comando identifica um ponto de identificação para transações. Ao utilizá-lo é possível reverter a transação até o ponto especificado. Isso faz com que não seja necessário reverter toda a transação desde o último COMMIT.

```
SAVEPOINT nome_savepoint;
```

O nome utilizado para identificar a transação deve ser exclusivo para as transações associadas.

Para que uma transação seja revertida até o ponto especificado pelo comando SAVEPOINT, utilizamos o comando:

```
ROLLBACK TO nome_savepoint;
```

Exemplo de utilização:

```
INSERT INTO AUTOR (CODIGO_AUTOR,NOME_AUTOR)
    VALUES ( 61, 'Zé Ramalho');
1 linha criada.
```



```
SAVEPOINT svp1;
Ponto de salvamento criado.

INSERT INTO MUSICA(CODIGO_MUSICA,NOME_MUSICA)
    VALUES (89,'Canção da América');
1 linha criada.

SAVEPOINT svp2;
Ponto de salvamento criado.

DELETE FROM MUSICA
    WHERE CODIGO_MUSICA NOT IN
    (SELECT CODIGO_MUSICA
    FROM FAIXA);
1 linha deletada.

ROLLBACK TO svp2;
Rollback completo.

SELECT * FROM MUSICA WHERE CODIGO_MUSICA = 89;

CODIGO_MUSICA   NOME_MUSICA
-------------   ----------------
89              Canção da América

SELECT * FROM AUTOR WHERE CODIGO_AUTOR = 61;

CODIGO_AUTOR    NOME_AUTOR
------------    ----------
61              Zé Ramalho
```

Observe que, como foi feito um ROLLBACK até o ponto SVP2, o autor permanece na tabela, pois o INSERT está antes do ponto SVP2. A música também está na tabela porque ela foi incluída antes do SVP2 e foi excluída depois desse ponto. Como voltamos à transação, apenas a inclusão da linha foi mantida. A exclusão não foi considerada.

Níveis de isolamento

O padrão SQL determina que se possa especificar quatro níveis de transações:

1. SERIALIZABLE: uma transação é totalmente isolada das outras transações. Caso a transação tenha comandos DML que tentem atualizar dados não gravados de outra transação, essa transação não será efetuada.

2. READ COMMITED: caso a transação utilize DML que precise do bloqueio de linhas que outras transações estão utilizando, a operação somente será concluída após a liberação da linha da outra transação.

3. REPEATABLE READ: os dados podem ser lidos mais de uma vez, e se outra transação tiver incluído ou atualizado linhas e estas forem gravadas no banco de dados entre uma e outra leitura dos dados, então os dados retornados da última busca serão diferentes dos dados da busca anterior. Esse efeito é conhecido como *fantasma*.

4. READ UNCOMMITED: conforme o nome diz, serão lidos conteúdos não gravados ainda pelo banco de dados (transações passíveis de ROLLBACK). Há um enorme risco nessas operações, visto que o usuário que está bloqueando a informação pode descartá-la.

O comando utilizado para determinar o nível de isolamento é:

```
SET TRANSACTION ISOLATION LEVEL nível;
```

Exemplo:

```
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
```

Observe que devemos utilizar esse comando antes de efetuar as transações no banco de dados.

É possível determinar a forma que as transações vão ocorrer no banco de dados. O padrão é permitir a leitura e a gravação dos dados, mas deixar o banco de dados em modo só de leitura, quando se extraem relatórios por exemplo, em que as transações jamais vão envolver bloqueios. Isso é interessante para permitir que mais usuários possam trabalhar concorrentemente.

Isso é feito utilizando o mesmo comando, mas com cláusulas específicas somente para leitura e leitura e gravação:

```
SET TRANSACTION READ ONLY;
SET TRANSACTION READ WRITE;
```

O padrão SQL não determina um comando para início ou fim de transações. Normalmente, isso é implícito assim que uma nova transação é iniciada na seção do banco de dados. Alguns bancos de dados, contudo, utilizam os comandos START WORK ou START TRANSACTION para determinar o início de uma transação.

Determinando quando CONSTRAINTs são verificadas

O padrão SQL determina que os bancos de dados tenham a habilidade de checar quando as CONSTRAINTs devem ser verificadas: em cada comando SQL ou no final da transação. Isso pode ser particularmente útil quando queremos realizar alterações que envolvam várias tabelas em cascata. Ao colocar o banco de dados em verificação, apenas no final da transação elimina-se o problema dessas modificações.

Para isso, utilizamos o seguinte comando:

```
SET CONSTRAINTS [MODE]
   {constraint|ALL}
   {IMMEDIATE|DEFERRED};
```

O padrão dos bancos de dados é IMMEDIATE. Quando especificamos uma *constraint*, apenas ela será afetada pelo comando, as demais permanecerão com o padrão do banco de dados.

Exercícios propostos

1. Crie um usuário no banco de dados.

2. Atribua direitos de conexão a esse usuário.

3. Atribua direitos de consulta na tabela comprador e vendedor.

4. Atribua direitos de inclusão e atualização nas tabelas oferta e imóvel.

5. Atribua direitos de exclusão nas tabelas cidade, Estado e bairro.

6. Revogue os direitos concedidos no exercício 5.

7. Altere sua sessão para que as transações sejam em série.

8. Altere sua sessão para que as constraints sejam verificadas apenas no final das transações.