GRANT e REVOKE – Definindo privilégios de acesso aos bancos de dados no MySQL

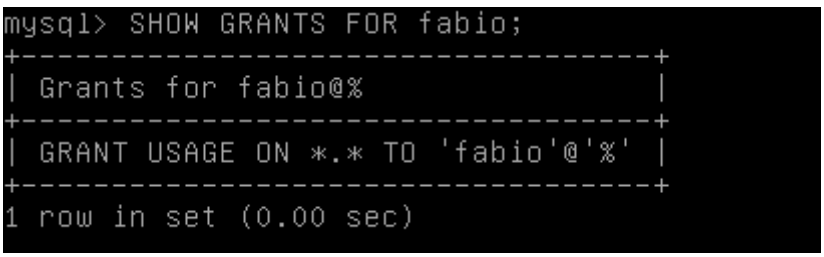
Nesta lição vamos mostrar como atribuir ou retirar privilégios de acesso aos bancos de dados para os usuários, usando as declarações **GRANT** e **REVOKE**.

No geral, usamos a declaração CREATE USER para criar usuários e a declaração GRANT para atribuir privilégios de acesso a eles. Também é possível criar um usuário e atribuir-lhe privilégios de uma vez com a própria declaração GRANT.

Visualizando os privilégios de um usuário

Para visualizarmos os privilégios de um usuário podemos usar a declaração **SHOW GRANTS**. Por exemplo, para ver os privilégios do usuário **fabio**:

```
SHOW GRANTS FOR fabio@localhost;
```



A saída do comando mostra que o usuário fabio possui o privilégio “USAGE” em todas as tabelas de todos os bancos de dados do sistema (*.*).

Na tabela a seguir temos um resumo dos privilégios mais comuns que um usuário de um banco de dados pode ter:

Privilégios para trabalhar com dados:	
Privilégio	***Descrição***
INSERT	Inserir dados em uma tabela
UPDATE	Atualizar dados em uma tabela
DELETE	Excluir dados de uma tabela
EXECUTE	Executar funções ou procedimentos armazenados
SELECT	Efetuar consultas em uma tabela

Privilégios para modificar a estrutura do banco de dados:	
Privilégio	***Descrição***
CREATE	Criar tabela ou banco de dados
ALTER	Modificar uma tabela
DROP	Excluir uma tabela ou um banco de dados
CREATE VIEWS	Criar exibições
TRIGGER	Criar ou excluir um trigger em uma tabela
INDEX	Criar ou excluir um índice
CREATE ROUTINE	Criar uma função ou um procedimento armazenado
ALTER ROUTINE	Alterar ou excluir uma função ou procedimento armazenado
Privilégios Administrativos	
Privilégio	***Descrição***
CREATE USER	Criar contas de usuários
SHOW DATABASES	Ver os nomes dos bancos de dados no servidor
SHUTDOWN	Desligar o servidor
RELOAD	Recarregar as tabelas que armazenam os privilégios dos usuários dos bancos de dados. Assim elas são atualizadas se tiverem sido modificadas.
Outros privilégios	
ALL	Todos os privilégios disponíveis em um determinado nível, exceto GRANT OPTION
GRANT OPTION	Permite dar privilégios a outros usuários
USAGE	Não altera privilégios; usado para tarefas administrativas na conta do usuário.

Níveis dos privilégios

No MySQL os privilégios são atribuídos em quatro níveis diferentes:

- **Global** – O usuário tem acesso a todas as tabelas de todos os bancos de dados
- **Database** – Esse privilégio dá ao usuário acesso a todas as tabelas de um banco de dados específico
- **Table** – O usuário tem aceso a todas as colunas de uma tabela específica em um banco de dados
- **Column** – O usuário possui acesso apenas a colunas especificadas em uma determinada tabela.

Armazenando informações sobre privilégios

O MySQL utiliza tabelas especiais denominada **grant tables** para armazenar informações sobre os privilégios dos usuários, em um banco de dados interno de nome **mysql**. A tabela a seguir detalha essas tabelas de privilégios:

Tabela	Descrição

user	Armazena nomes e senhas de todos os usuários do servidor. Também armazena os privilégios globais que são aplicados a todos os bancos de dados do servidor.
db	Armazena privilégios dos bancos de dados
tables_priv	Armazena privilégios das tabelas
columns_priv	Armazena privilégios de colunas
procs_priv	Armazena privilégios de acesso a funções e stored procedures (procedimentos armazenados).

Usando a declaração GRANT para atribuir privilégios

Sintaxe:

```
GRANT lista_privilégios
ON [nome_banco.]tabela
TO usuário1 [IDENTIFIED BY 'senha1'],
usuário2 [IDENTIFIED BY 'senha2'] ...
[WITH GRANT OPTION]
```

Vamos a alguns exemplos de uso:

1 – Garantir acesso a um usuário de nome **julia**, sem privilégios:

```
GRANT USAGE
ON *.*
TO julia@localhost IDENTIFIED BY '1234';
```

Para verificar:

```
SHOW GRANTS FOR julia@localhost;
```

2 – Dar privilégios globais a um usuário de nome alexandre:

```
GRANT ALL
ON *.*
TO alexandre IDENTIFIED BY '1234'
WITH GRANT OPTION
```

Para verificar:

```
SHOW GRANTS FOR alexandre;
```

3 – Dar privilégios específicos para execução de comandos DML em todas as tabelas do banco db_biblioteca ao usuário ana:

```
GRANT SELECT, INSERT, UPDATE, DELETE
ON db_biblioteca.*
TO ana@localhost;
```

Para verificar:

```
SHOW GRANTS FOR ana@localhost;
```

4 – Dar todos os privilégios no banco db_biblioteca à usuária ana:

```
GRANT ALL
ON db_biblioteca.*
TO ana@localhost;
```

Para verificar:

```
SHOW GRANTS FOR ana@localhost;
```

5 – Garantir privilégios de inserção e atualização de registros e efetuar consultas na a tabela **tbl_autores** do banco de dados **db_biblioteca** ao usuário julia:

```
GRANT SELECT, INSERT, UPDATE
ON db_biblioteca.tbl_autores
TO julia@localhost;
```

Para verificar:

```
SHOW GRANTS FOR julia@localhost;
```

6 – Garantir o privilégio de consultar nomes e sobrenomes e alterar somente nomes dos autores (coluna nome-autor) da tabela tbl_autores do banco db_biblioteca ao usuário fabio:

```
GRANT SELECT (nome_autor, sobrenome_autor), UPDATE (nome_autor)
ON db_biblioteca.tbl_autores
TO fabio@localhost;
```

Para verificar:

```
SHOW GRANTS FOR fabio@localhost;
```

Revogando privilégios com a declaração REVOKE

Podemos revogar (retirar) privilégios dos usuários usando a declaração **REVOKE**.

Sintaxe:

```
REVOKE lista_privilégios
ON objeto
FROM usuário1, usuário2, ...;
```

Exemplos:

1 – Vamos revogar o privilégio de exclusão de dados no banco db_biblioteca à usuária ana:

```
REVOKE DELETE
ON db_biblioteca.*
FROM ana@localhost;
```

Para verificar:

```
SHOW GRANTS FOR ana@localhost;
```

2 – Retirando o privilégio de atualização da coluna nome_autor do banco db_biblioteca, na tabela de autores, do usuário fabio:

```
REVOKE UPDATE (nome_autor)
ON db_biblioteca.tbl_autores
FROM fabio@localhost;
```

Conferindo:

```
SHOW GRANTS FOR fabio@localhost;
```

3 – Remover todos os privilégios em todos os bancos de dados dos usuários alexandre e ana:

```
REVOKE ALL, GRANT OPTION
FROM alexandre, ana@localhost;
```

Para verificar:

```
SHOW GRANTS FOR alexandre;
SHOW GRANTS FOR ana@localhost;
```

Observação: A declaração REVOKE não exclui um usuário do sistema; para isso, use a declaração **DROP USER**.