SQL Server 安全管理

数据库的安全性是指保护数据以防止因不合法的使用而造成的数据的泄密和破坏

1.SQL Server 的安全性机制

SQL Server 的安全体系结构可以划分为以下 4 个层次：

1.1 客户操作系统的安全性

在使用客户计算机通过网络实现对 SQL Server 服务器的访问时，用户首先要获得计算机操作系统的使用权限。

1.2SQL server 的服务器安全性

SQL Server 服务器安全性建立在控制服务器登录账号和密码的基础上。

管理和设计合理的登录方式是数据库管理员（DBA）的重要任务，是 SQL Server 安全体系中 DBA 可以发挥主动性的第一道防线。

SQL Server 事先设计了许多固定的服务器角色，用来为具有服务器管理员资格的用户分配使用权限。

拥有固定服务器角色的用户可以拥有服务器级别的管理权限。

1.3 数据库使用的安全性

用户将接受的第三次安全性检验就是数据库的安全性检查。

建议用户在建立新的登录账号时，最好不要讲默认的数据库设置为 master 数据库，而是应该根据用户实际需要，将默认的数据库设置在具有实际操作意义的数据库上。

SQL Server 中也包含一些固定服务器角色，以方便对数据库访问权限的管理。

1.4 数据库对象使用的安全性

在创建数据库对象时，SQL Server 自动把该数据库对象拥有权赋予该对象的创建者。

当一个非数据拥有者想访问数据库里的对象时，必须事先由数据库拥有者赋予用户对指定对象执行特定操作的权限。

一般来说，为减少管理的开销，在对象级安全管理上应该在大多数场合赋予数据库用户以广泛的权限，然后在针对实际情况在某些敏感的数据上实施具体的访问控制权限控制。

每个安全层次好像是一道门，如果门没有上锁，或者用户拥有开门的权利，则永和可以通过这道大门到达下一个安全等级。如果通过了所有的大门，用户就可以实现对数据的访问了。

2.SQL Server 验证模式

用户要访问 SQL Server 数据库中的数据，必须经过三个认证过程：

身份验证

—验证用户是否具有连接到服务器的资格

安全账户认证是用来确认登录 SQL Server 的用户的登录账号和密码的正确性，由此来验证其是否具有连接 SQL Server 的权限。提供了两种验证模式：

一：windows 身份认证模式

二：混合身份认证模式

验证用户是否是数据库的合法用户

—验证用户是否具有对数据库的访问权

验证用户是否具有操作许可

—验证用户是否具有对数据库中的数据和对象的操作权限

3.管理 SQL Server 登录账户

3.1windows 身份验证码模式

创建 windows 登录账户

```
CREATE LOGIN LoginName FROM WINDOWS

[WITH DEFAULT_DATABASE=DatabaseName]
```

拒绝连接：

```
DENY CONNECT SQL TO LoginName
```

删除用户：

```
DROP LOGIN LoginName
```

3.2SQL Server 身份验证模式

创建 SQL Server 登录账户

```
CREATE LOGIN LoginName

[

  WIRH PASSWORD = 'password'

  [MUST_CHANGE]

  [,DEFAULT_DATABASE=DatabaseName]

  [,CHECK_EXPIRATION={ON|OFF}]

  [,CHECK_POLICY={ON|OFF}]
```

```
]
```

例子：

```
#创建一个 SQL Server 登录账户 sql2
CREATE LOGIN sql2 WITH PASSWORD='sql2'
GO
#修改登录账户密码
ALTER LOGIN sql2 WITH PASSWORD='123456'
GO
#删除 SQL Server 登录账户
DROP LOGIN Sql2
GO
```

4.管理固定服务器角色

SQL Server 管理者可以将某些用户设置为某一角色，这样支队角色进行权限设置便可以实现对所有用户权限进行设置，大大减少了管理员的工作量。SQL Server 提供了两种角色:服务器角色和数据库角色。

服务器角色：用户管理 SQL Server 服务器级别的权限

数据库角色：用户管理数据库级别的权限

4.1 固定服务器角色

固定服务器角色是指根据 SQL Server 的管理任务，以及这些任务相对重要性

等级来把具有 SQL Server 管理职能的用户划分为不同的用户组，每一组所具有的的管理 SQL Server 的权限都是 SQL Server 内置的，即不能对其进行添加、修改和删除权限，只能向其中加入用户或者其他角色。

4.2 固定服务器角色及权限

拥有 SQLServer 所有的权限许可。

服务器管理员 serveradmin：管理 SQLServer 服务器端的设置。

磁盘管理员 diskadmin：管理磁盘文件。

进程管理员 processadmin：管理 SQLServer 系统进程。

安全管理员 securityadmin：管理和审核 SQLServer 系统登录。

安装管理员 setupadmin：增加、删除连接服务器，建立数据库复制以及管理扩展存储过程。

数据库创建者 dbcreator：创建数据库，并对数据库进行修改。

Bulkadmin：执行 BULKINSERT 语句

4.3 固定服务器角色的分配与撤销

可以使用系统存储过程 sp_helpsrvrolemember 查看摸个固定服务器角色被分配给了那些 SQL Server 服务器登录账户，使其拥有相应的服务器级操作权限。分配角色的语法格式如下：

```
EXEC sp_helpsrvrolemember 'RoleName'

#将 sysadmin 角色分配给 SQL Server 服务器登录账户 Sql1

EXEC sp_addsrvrolemember 'Sql1','sysadmin'

GO
```

```
#查看 sysadmin 角色下包括那些 SQL Server 服务器登录账户
EXEC sp_helpsrolemember 'sysadmin'
GO
#撤销固定服务器角色的分配
EXEC sp_dropsrvrolemember 'LoginName','RoleName'
#例如撤销 sql1 分配的角色 sysadmin
EXEC sp_dropsrvrolemember 'sql1','sysadmin'
GO
```

特殊的 SQL Server 登录账户 sa

sa 账户拥有服务器级别最高的权限管理，可以执行服务器范围内的所有操作。

5.数据库安全管理

5.1 管理数据库用户

5.1.1 创建数据库用户

可以使用 create user 语句创建数据库用户，其基本的语法如下

```
CREATE USER USERNAME [FOR LOGIN LOGINNAME]
```

例：在 student 数据库中为登录账户 sql1 创建用户 sqlUser1

```
USE Student
GO
CREATE USER SqlUser1 FOR LOGIN Sql1;
```

5.1.2 查看数据库用户

例：查看数据库 Student 中的用户信息

```
USE Student
GO
SELECT *
FROM sys.database_principals
GO
```

5.1.3 修改和删除数据库用户

可以使用 ALTER USER 语句修改数据库用户

例：将数据库用户 SqlUser1 的名字改为 Sql1

```
USE Student
GO
ALTER USER sqlUSER1 WITH NAME = Sql1
Go
```

可以使用 DROP USER UserName 语句删除指定的数据库用户

5.2 数据库对象权限管理

权限用来指定授权用户可以使用的数据库对象和这些授权用户可以对这些数据库对象执行的操作。用户在登录到 SQL Server 之后，其用户账号所归属的 NT 组或角色被赋予的权限（许可）决定了该用户能够对那些数据库对象执行那种操作以及能够访问、修改那些数据。在每个数据库中用户的权限独立于用户账

号和用户在数据库中的角色,每个数据库独有自己独立的权限系统,在 SQL Server 中包括三种类型的权限:

1.对象权限

表示对特定的数据库对象，即表、视图、字段和存储过程的操作许可，他据决定了能对表、视图等数据库对象执行那些操作。（相当于 DML 语句权限）

2.语句权限

表示对数据库的操作许可，也就是说，创建数据库或者创建数据库中的其他内容所需要的许可类型称为语句许可（相当于 DDL 的语言权限）

3.隐含权限

是系统安装以后有些用户和角色不必授权就有许可。

5.2.1 权限管理

权限的管理包含三个内容:

授予权限：允许用户或角色居于某种操作权

收回权限：不允许用户或角色居于魔种操作权，或者收回曾经授予的权限。

拒绝访问：拒绝某用户或角色具有某种操作权。即使用户或角色用于继承而获得这种操作权，也不允许该用户执行相应的操作。

例：使用 Transaction_SQl 语句管理对象权限

Transaction_SQL 语句使用 grant、 revoke 和 deny 三种语句来实现管理对象权限

```
GRANT:用户授权

REVOKE:用于收回权限
```

```
DENY:用于拒绝权限
```

授权-grant 语句

grant 语句的格式为：

grant 对象权限名 [,....] on {表名|视图名|存储过程名} To {数据库用户名|用户角色名} [,.....]

实例：

```
grant select on authors to user1
```

收回权限-REVOKE 语句

REVOKE 语句的格式为：

REVOKE 语句的格式为：

REVOKE 对象权限名 [,...] ON {表名|视图名|存储过程名}

FROM {数据库用户名|用户角色名} [,....]

实例：

```
REVOKE SELECT ON authors FROM user1
```

拒绝访问语句-DENYyuju

DENY 语句的格式为：

DENY 对象权限名 [,...] ON {表名|视图名|存储过程名}

To {数据库用户名|用户角色名} [,...]

实例：

```
DENY UPDATE ON authors To user1
```

例：使用 Transaction_SQL 语句管理语句权限

Transaction_SQL 语句也是使用 grant、revoke 和 deny 三种语句来实现语句管理权限

GRANT 用户授权；

REVOKE 用户收回权限；

DENY 用于拒绝权限。

授权-Grant 语句

grant 语句的格式为：

grant 语句授权名 [,…] To {数据库用户名|用户角色名}[,….]

实例：

```
grant create table to user1;
```

收回权限-revoke 语句

revoke 语句的格式为：

revoke 语句权限名 [,…] from {数据库用户名|用户角色名}[,….]

实例：

```
revoke create table,create view from user3;
```

拒绝权限-DENY 语句

DENY 语句的格式为：

DENY 语句权限名 [,…] To {数据库用户名|用户角色名} [,…]

实例：

```
DENY CREATE VIEW TO user1;
```

5.3 管理数据库角色

数据库角色是为某一用户或某一组用户授予不同级别的管理或者访问数据库以及数据库对象的权限，这些权限是数据库专有的，并且还可以使一个用户具有属于同一数据库的多个角色：即固定的数据库角色和用户自定义的数据库角色。

5.4 固定数据库角色

固定的数据库角色用来提供最基本的数据库权限管理

public:维护全部默认权限

所有数据库用户都属于 public 角色。特点：

public 角色存在于每一个数据库中，包括系统数据库和用户数据库，而且不能被删除

因为所有的数据库用户都属于该角色，所以不能直接将该角色分配给任何用户

public 角色在初始状态没有任何权限，但可以为该角色分配权限

因为所有的数据库都属于该角色，所以为该角色授权时，实际是为所有数据库用户授权。

db_owner:数据库的所有者，可以对所拥有的数据库执行任何的操作

db_accessadmin:可以增加或者删除数据库用户，工作组和角色

db_addladmin:可以增加、删除和修改数据库中任何对象

db_securityadmin:执行语句权限和对象权限

db_backupoperator:可以备份和恢复数据库

db_datareader:能且仅能对数据库中的任何表执行 select 操作，从而读取所有表的信息。

db_datawriter:能够增加、修改和删除表中的数据，但不能进行 select 操作。

db_denydatareader:不能读取数据库中任何表中的数据

db_denydataewriter:不能对数据库中任何表执行增加、修改和删除数据操作

用户自定义数据库角色

```
USE Student

Go

Create ROLE user Role1

Go
```

为数据库用户分配和撤销角色

为用户分配角色的语法

```
EXEC ap_addrolemember 'roleName','securityaccount'

#例：将固定数据库角色 db_ower 分配给 student 的 sql

USE Student

Go

EXEC sp_addrolemember 'db_ower','sql1'

Go
```