第5章

数据库安全技术

安全性问题不是数据库系统所独有的，所有计算机系统都有这个问题。只是在数据库系统中大量数据集中存放，且可为许多用户所共享，从而使得安全性问题更为突出。数据库的安全性已成为评价数据库系统性能的一个重要指标。

数据库的安全性是指保护数据库以防止因不合法的使用而造成数据的泄露、更改或破坏。本章介绍数据库安全性控制的各种技术，并详细介绍 SQL Server 2014 中的安全机制。

5.1 安全性控制技术概述

数据库系统的安全性和计算机系统的安全性（包括计算机硬件、操作系统、网络系统的安全性）是紧密联系、相互支持的。

5.1.1 计算机系统的三类安全性问题

计算机系统的安全性是指为计算机系统建立和采取的各种安全保护措施，以保护计算机系统中的硬件、软件及数据，防止其因偶然或恶意的原因使系统遭到破坏或丢失，数据遭到更改或泄露等。

影响计算机系统安全性的因素很多，不仅涉及计算机系统本身的技术问题、管理问题，还涉及法学、犯罪学、心理学的问题。其内容包括了计算机安全理论与策略、计算机安全技术、安全管理、安全评价、安全产品以及计算机犯罪与侦察、计算机安全法律、安全监察等。概括起来，计算机系统的安全性问题可分为三大类，即技术安全类、管理安全类和政策法律类。

（1）技术安全是指采用具有一定安全性的硬件、软件来保护计算机系统中的硬件、软件及数据。当计算机系统受到无意或恶意的攻击时仍能保证系统正常运行，保证系统内的数据不丢失、不泄露。

（2）管理安全是指防止因管理不善所导致的计算机设备和数据介质的物理破坏或丢失，防止软硬件意外故障以及场地的意外事故。

（3）政策法规是指政府部门建立的有关计算机犯罪、数据安全保密的法律道德准则和政策法规、法令等。

这里只讨论技术安全。

5.1.2 安全标准简介

计算机及其信息安全技术方面有一系列的安全标准，其中有重要影响的是 1985 年美国国防部（DoD）颁布的《DoD 可信计算机系统评估标准》（Trusted Computer System

Evaluation Criteria，TCSEC）和 1991 年美国国家计算机安全中心（NCSC）颁布的《可信计算机系统评估标准关于可信数据库系统的解释》（Trusted Database Interpretation，TDI）。TDI 将 TCSEC 扩展到数据库管理系统，定义了数据库系统设计与实现中需要满足和用以进行安全性级别评估的标准。

根据系统对各项指标的支持情况，TCSEC/TDI 将系统划分为 DCBA 4 组，D、C1、C2、B1、B2、B3 和 A1 从低到高 7 个等级。按系统可靠或可信程度逐渐增高，各安全级别之间偏序向下兼容，较高安全等级提供的安全保护包含较低等级的所有保护要求，同时提供更完善的保护。7 个安全等级的基本要求如下。

（1）D 级为最低级别，提供最小保护，凡不符合更高标准的系统，统统归于 D 组。如 DOS 就是操作系统中安全标准为 D 的典型例子，它具有操作系统的基本功能，如文件系统、进程调度等，但在安全性方面几乎没有专门的机制来保障。

（2）C1 级只提供了非常初级的自主安全保护。能够实现用户与数据的分离，进行自主存取控制（DAC），保护或限制用户权限的传播。

（3）C2 级是安全产品的最低档次，提供受控的存取保护。将 C1 级的 DAC 进一步细化，以个人身份注册负责，并实施审计和资源隔离。如 Windows 2000，Oracle 7 等。

（4）B1 级提供标记安全保护。对系统的数据加以标记，并对标记的主体和客体实施强制存取控制（MAC）。B1 级较好地满足了大型企业或一般政府部门对于数据的安全需求，这一级别的产品才被认为是真正意义上的安全产品。满足 B1 级别的产品出售时，允许冠以“安全”（Security）或“可信的”（Trusted）字样。

（5）B2 级提供结构化保护。建立形式化的安全策略模型并对系统内的所有主体和客体实施 DAC 和 MAC。

（6）B3 级提供安全域保护。要求可信任的运算基础必须满足访问监控器的要求，审计跟踪能力更强，并提供系统恢复过程。

（7）A1 级为验证设计。提供 B3 级保护的同时给出系统的形式化设计说明和验证，以确保各安全保护的真正实现。

在 TCSEC 推出后，不同国家都开始启动开发建立在 TCSEC 概念上的评估准则。如欧洲的信息技术安全评估准则（ITSEC）、加拿大的可信计算机产品评估准则（CTCPEC）、美国的信息技术安全联邦标准（FC）草案等。

为了准确地评估和测定计算机系统的安全性能，规范和指导计算机系统的生产，满足全球 IT 市场上互认标准化安全评估结果的需要，1993 年 CTCPEC、FC、TCSEC 和 ITSEC 的发起组织开始联合行动，解决原标准中概念和技术上的差异，将各自独立的准则集合成一组单一的、能被广泛使用的 IT 安全准则，这一行动被称为 CC（Common Criteria）项目。项目发起组织的代表建立了专门的委员会来开发 CC 通用准则，历经多次讨论和修改，CC V2.1 版于 1999 年被 ISO 采用为国际标准，2001 年被我国采用为国家标准。目前 CC 已经基本取代了 TCSEC，成为评估信息产品安全性的主要标准。

和早期的评估准则相比，CC 具有结构开发通用、表达方式通用等特点。它把对信息产品的安全要求分为安全功能要求和安全保证要求，前者用以规范产品和系统的安全行为，后者解决如何正确有效地实施这些功能。CC 在安全保证要求部分定义了 7 种评估保证级别，从 EAL1 至 EAL7，按保证程度逐渐增高。粗略而言，TCSEC 的 C1 和 C2 级分别相当于

EAL2 和 EAL3；B1、B2 和 B3 分别相当于 EAL4、EAL5 和 EAL6；A1 对应于 EAL7。

5.1.3 数据库安全性控制概述

在网络环境下，数据库的安全体系涉及三个层次：网络系统层、操作系统层和数据管理系统层。它们与数据库的安全性紧密联系并逐层加强，从外到内保证数据的安全。在规划和设计数据库的安全性时，要综合每一层的安全性，使三层之间相互支持和配合，提高整个数据库系统的安全性。

数据库安全性控制是指为防止数据库的不合法使用和因偶然或恶意的原因，使数据库中数据遭到泄露、更改或破坏等所采取的各种技术和安全保护措施的总称。

这里的讨论只限于数据库管理系统这一层次。

1. 用户标识和鉴别

在一个多用户的数据库系统中，用户的标识(Identification)和鉴别(Authentication)是系统安全控制机制中最重要、最外层的安全保护措施。其方法是由系统提供一定的方式让用户标识自己的身份。每当用户要求进入系统时，系统首先根据输入的用户标识进行身份的鉴定，只有鉴定合法的用户才能进入系统。鉴别用户最常用的方法是用一个用户名(用户标识符)来标识用户，用口令来鉴别用户身份的真伪。

口令简单易行，但容易被人窃取。一种可行的方法是，每个用户预先约定好一个计算过程或者函数，鉴别用户身份时，系统提供一个随机数，用户根据自己预先约定的计算过程或者函数进行计算，系统根据用户计算结果是否正确鉴别用户身份。用户标识和鉴别可以重复多次。动态口令是目前较为安全的一种鉴别方式。这种方式的口令是动态变化的，每次鉴别时均需使用动态产生的新口令(如短信密码)登录数据库管理系统，即采用一次一密的方法。与静态口令鉴别相比，这种方式增加了口令被窃取或破解的难度，安全性相对高一些。

2. 存取控制

数据库安全最重要的是确保只授权给有资格的用户访问数据库的权限，同时令所有未被授权的人员无法接近数据。这一点主要是通过数据库系统的存取控制机制实现的。存取控制机制主要包括两部分。

(1) 定义用户权限。用户对某一数据库对象的操作权力称为权限。某个用户对某一数据库对象应该具有何种权限，这是管理问题和政策问题而不是技术问题，DBMS 的功能是保证这些决定的执行。为此，DBMS 必须提供适当的语言来定义用户权限，这些定义经过编译后存放在系统的数据字典中，被称为安全规则或授权规则。

(2) 合法权限检查。每当用户发出存取数据库的操作请求后(请求一般应包括操作类型、操作对象和操作用户等信息)，DBMS 查找数据字典，根据安全规则进行合法权限检查。若用户的操作请求超出了定义的权限，系统将拒绝执行此操作。

定义用户权限和合法权限检查一起组成了 DBMS 的安全子系统。

常用的存取控制方法有 C2 级中的自主存取控制(Discretionary Access Control，DAC)和 B1 级中的强制存取控制(Mandatory Access Control，MAC)两种。

在自主存取控制方法中，用户对于不同的数据库对象有不同的存取权限，不同的用户对同一对象也有不同的权限，而且用户还可将其拥有的存取权限转授给其他用户。自主存取

控制非常灵活，但可能存在数据的“无意泄漏”。因为用户对数据的存取权限是“自主”的，用户可以自由地决定是否将数据的存取权限授予别人，也可以自由地决定是否将“授权”的权限授予别人。造成这一问题的根本原因是这种机制仅仅通过对数据的存取权限来进行安全控制，而数据本身并无安全性标记。要解决这一问题，就需要对系统控制下的所有主客体（主体可理解为用户，客体可理解为数据库对象）实施强制存取控制策略。

在强制存取控制方法中，每一个数据库对象被标以一定的密级，每一个用户也被授予某一个级别的许可证。对于任意一个对象，只有具有合法许可证的用户才可以存取。强制存取控制保证更高程度的安全性，适用于那些对数据有严格且固定的密级分类的部门，例如军事部门或政府部门。

目前大型的 DBMS 一般都支持 DAC，有些 DBMS 同时还支持 MAC。本章稍后将讨论 DAC。

3. 视图机制

通过为不同的用户定义不同的视图，可以把数据对象限制在一定的范围内。也就是说，通过视图机制把要保密的数据对无权存取的用户隐藏起来，从而自动地对数据提供一定程度的安全保护。

视图机制间接地实现支持存取谓词的用户权限定义。例如，在某大学中假定赵亮老师只能检索计算机专业学生的信息，专业负责人朱伟强具有检索和增删改计算机专业学生信息的所有权限。这就要求系统能支持“存取谓词”的用户权限定义。在不直接支持存取谓词的系统中，可以先建立计算机专业的学生视图 CS_S，然后在视图上进一步定义存取权限。

例 5.1 把检索计算机专业学生信息的权限授予赵亮，把对计算机专业学生信息的所有操作权限授予朱伟强。

```
CREATE VIEW CS_S
AS SELECT *
   FROM S
   WHERE Major = '计算机'
GRANT SELECT ON CS_S TO 赵亮
GRANT ALL PRIVILEGES ON CS_S TO 朱伟强
```

4. 审计

用户标识与鉴别、存取控制仅是安全性控制的一个重要方面而不是全部。为了使 DBMS 达到一定的安全级别，还需要在其他方面提供相应的支持。例如，按照 TDI/TCSEC 标准中安全策略的要求，“审计”功能是 DBMS 达到 C2 以上安全级别必不可少的一项指标。

因为任何系统的安全保护措施都不是完美无缺的，蓄意盗窃、破坏数据的人总是想方设法打破控制。审计功能把用户对数据库的所有操作自动记录下来存放在审计日志中，DBA 可以利用审计跟踪信息，重现导致数据库现有状况的一系列事件，找出非法存取数据的人、时间和内容等。

审计通常是很费时间和空间的，所以 DBMS 往往都将其作为可选特征，允许 DBA 根据应用对安全性的要求，灵活地打开或关闭审计功能。审计功能一般主要用于安全性要求较高的部门。

5. 数据加密

对于高度敏感的数据，如财务数据、军事数据、国家机密，除了以上安全性控制措施外，

还可以采用数据加密技术。数据加密是防止数据库中的数据在存储和传输中失密的有效手段。加密的基本思想是根据一定的算法将原始数据(也即明文)变换为不可直接识别的格式(也即密文),从而使得不知道解密算法的人无法获知数据的内容。

加密方法主要有两种。一种是替换方法,该方法使用密钥将明文中的每一个字符转换为密文的一个字符,另一种是置换方法,该方法仅将明文中的字符按不同的顺序重新排列。单独使用这两种方法中的任意一种都是不够安全的,但是将这两种方法结合起来就能提供相当高的安全程度。采用这种结合算法的典型例子是美国1977年制定的官方加密标准,数据加密标准(Data Encryption Standard,DES),详细的讨论请参考相关文献。

由于数据加密与解密也是比较费时的操作,而且数据加密与解密程序会占用大量系统资源,因此数据加密功能通常也作为可选特征,允许用户自由选择,只对高度机密的数据加密。

5.2 用户管理和角色管理

安全控制首先是用户管理,DBMS通过用户账户对用户的身份进行识别,从而完成对数据资源的控制。

5.2.1 用户管理

在SQL Server中,有登录名(Login Name)和数据库用户(Database User)两个概念。登录名用于验证用户是否有权限连接到SQL Server服务器,数据库用户用于验证用户登录服务器后是否有对服务器上的某个数据库进行操作的权限。用户必须拥有和自己登录名对应的数据库用户才可以对某个数据库进行操作,这样增强了数据库的安全性,避免了一个用户在登录到服务器后可以对服务器上的所有数据库进行操作。一个登录名可以是多个数据库的用户。DBA有权进行登录名和数据库用户的管理。

1. 创建登录名

在T-SQL语言中,创建登录名用CREATE LOGIN语句,其格式如下:

```
CREATE LOGIN <登录名> WITH PASSWORD = 'password'
                    [ , DEFAULT_DATABASE = <数据库名> ]
```

说明:在SQL Server中有4种类型的登录名,即SQL Server登录名、Windows登录名、证书映射登录名和非对称密钥映射登录名,这里只介绍SQL Server登录名。有一个特殊的登录名sa(即系统管理员),他拥有操作SQL Server系统的所有权限,该登录名不能被删除。数据库名是指定将指派给登录名的默认打开的数据库,如果缺省该选项,则默认打开的数据库将设置为master。

例5.2 创建SQL Server登录名carl,密码为1234,默认打开的数据库为master。

```
CREATE LOGIN carl WITH PASSWORD = '1234'
```

2. 删除登录名

在T-SQL语言中,删除登录名用DROP LOGIN语句,其格式如下:

```
DROP LOGIN <登录名>
```

例 5.3 删除 SQL Server 登录名 carl。

```
DROP LOGIN carl
```

说明：删除登录名，并不删除该登录名在各数据库中对应的数据库用户。

3. 创建数据库用户

前面已经提到，用户通过登录名连接到(即登录到)服务器后，并不表示该用户具有对服务器上的数据库进行操作的权限，还需要给这个登录名授予访问某个数据库的权限。这需要在所要访问的数据库中为该登录名创建一个数据库用户。

在 SQL Server 中，每个数据库中都有两个特殊的数据库用户 dbo 和 guest。dbo 代表数据库拥有者，sysadmin 服务器角色成员都被自动映射成 dbo 用户，dbo 用户默认是数据库角色 db_owner 的成员，可以对数据库进行所有操作。guest 用户是所有没有属于自己的用户账号的登录名的默认数据库用户，通过 guest 用户账号可以操作对应的数据库(当然前提是 guest 用户拥有了对该数据库操作的权限)。

在 T-SQL 语言中，在当前数据库中创建数据库用户用 CREATE USER 语句，其格式如下：

```
CREATE USER <用户名> [ { FOR | FROM } LOGIN <登录名> ]
                    [ WITH DEFAULT_SCHEMA = <架构名> ]
```

说明：登录名是新建数据库用户所对应的登录名，如果缺省该选项，则新建数据库用户将被映射到同名的 SQL Server 登录名。架构名是新建数据库用户的默认架构，如果缺省该选项，则将使用 dbo 架构作为其默认架构(Schema 在 SQL Server 中称为架构)。

例 5.4 在数据库 test2 中为 SQL Server 登录名 carl 创建数据库用户 carluser，默认架构为 dbo。

```
USE test2
GO
CREATE USER carluser FOR LOGIN carl
```

4. 删除数据库用户

在 T-SQL 语言中，删除当前数据库中的数据库用户用 DROP USER 语句，其格式如下：

```
DROP USER <用户名>
```

例 5.5 删除数据库 test2 中的数据库用户 carluser。

```
USE test2
GO
DROP USER carluser
```

5.2.2 角色管理

角色是被命名的一组与数据库操作相关的权限，角色是权限的集合。可以为一组具有相同权限的数据库用户创建一个角色，所以也可以说角色是具有相同权限的数据库用户组。对一个角色授权或收回权限适用于该角色的所有成员，因此使用角色来管理权限可以简化授权的过程。例如在大学中可以建立一个“全日制本科学生”角色，然后给这个角色授予适

当的权限。当新生入学后,可以将新生添加为该角色的成员;而当学生毕业离校后,可以将他们从这个角色中删除。这样就不必在每个学生入学或毕业时,反复授权或收回权限。权限在用户成为角色成员后自动生效。DBA 有权进行角色的管理。

1. 创建角色

在 T-SQL 语言中,在当前数据库中创建数据库角色用 CREATE ROLE 语句,其格式如下:

```
CREATE ROLE <角色名> [ AUTHORIZATION <所有者名> ]
```

说明:所有者名是即将拥有新角色的数据库用户或角色,如果缺省该选项,则执行 CREATE ROLE 的用户将拥有该角色。

例 5.6 在数据库 test2 中创建数据库角色 undergraduate,所有者为 dbo。

```
USE test2
GO
CREATE ROLE undergraduate AUTHORIZATION dbo
```

2. 为角色添加成员

在 SQL Server 中,每个数据库中都有一个特殊的 public 数据库角色,数据库中的每个用户都是该角色的成员。在 SQL Server 2014 中,可以使用系统存储过程 sp_addrolemember 为某个数据库角色添加成员(数据库用户或数据库角色),其格式如下:

```
EXEC sp_addrolemember <角色名> , <成员名>
```

说明:使用 sp_addrolemember 添加到角色中的成员会继承该角色的权限。角色不能将自身包含为成员(间接地循环定义也无效),也不能向角色中添加 SQL Server 固定角色或 dbo 用户。

例 5.7 为数据库角色 undergraduate 添加数据库用户 carluser。

```
EXEC sp_addrolemember undergraduate , carluser
```

3. 从角色删除成员

在 SQL Server 2014 中,如果某个数据库角色中的某个成员(数据库用户或数据库角色)不再担当该角色,可以使用系统存储过程 sp_droprolemember 从该数据库角色中将其删除,其格式如下:

```
EXEC sp_droprolemember <角色名> , <成员名>
```

说明:使用 sp_droprolemember 删除某个角色的成员后,该成员将失去作为该角色的成员身份所拥有的任何权限。不能删除 public 角色的成员,也不能从任何角色中删除 dbo 用户。

例 5.8 从数据库角色 undergraduate 中删除数据库用户 carluser。

```
EXEC sp_droprolemember undergraduate , carluser
```

4. 删除角色

在 T-SQL 语言中,从当前数据库中删除数据库角色用 DROP ROLE 语句,其格式如下:

```
DROP ROLE <角色名>
```

说明：无法从当前数据库中删除拥有成员的数据库角色。若要删除有成员的角色，必须首先删除角色的成员。不能使用 DROP ROLE 删除固定角色。

例 5.9 删除数据库 test2 中的数据库角色 undergraduate。

```
USE test2
GO
DROP ROLE undergraduate
```

5.3 权限管理

数据库安全最重要的是确保只授权给有资格的用户访问数据库的权限，同时令所有未被授权的人员无法接近数据。这一点主要是通过数据库系统的存取控制机制实现的。目前大型的 DBMS 一般都支持自主存取控制，SQL 语言提供了 GRANT 语句和 REVOKE 语句来支持自主存取控制。

用户权限由数据库对象和操作类型两个要素组成。定义一个用户的存取权限就是定义这个用户可以在哪些数据库对象上进行哪些类型的操作。定义用户的存取权限称为授权(Authorization)。在 SQL Server 2014 中，数据库对象有许多种类，其中主要有模式对象(如基本表 TABLE、视图 VIEW 等)和数据对象(如基本表或视图、基本表或视图中的某个列)两类。

5.3.1 授予权限

1. 授予模式对象权限

在 SQL Server 2014 中，要创建数据库、基本表或视图，用户必须拥有执行相应语句的权限。例如，如果一个用户要能够在数据库中创建基本表，则应该向该用户授予 CREATE TABLE 语句权限。在 T-SQL 语言中，授予模式对象权限的语句格式如下：

```
GRANT { ALL [ PRIVILEGES ] | <权限> [, …n] }
TO <用户名或角色名> [, …n] [ WITH GRANT OPTION ]
```

说明：DBA 有权执行该 GRANT 语句。WITH GRANT OPTION 表示该用户(或角色)还可以向其他用户(或角色)授予所指定的权限。模式对象的操作类型有 CREATE TABLE、CREATE VIEW 等(详细请参见 SQL Server 2014 提供的联机丛书)。操作类型 ALL 不代表所有可能的权限，授予 ALL 权限等同于授予 CREATE DATABASE、CREATE TABLE、CREATE VIEW、CREATE DEFAULT、CREATE RULE、CREATE FUNCTION、CREATE PROCEDURE、BACKUP DATABASE 和 BACKUP LOG 权限。

例 5.10 授予用户 carluser 在数据库 test2 中创建基本表和视图的权限。

```
GRANT CREATE TABLE, CREATE VIEW TO carluser
```

2. 授予数据对象权限

在 SQL Server 2014 中，要查询或者更新基本表或视图中的数据，用户在该基本表或视图上必须先拥有相应的权限。在 T-SQL 语言中，授予数据对象权限的语句格式如下：

```
GRANT { ALL [ PRIVILEGES ] | <权限> [ ( <列名> [, …n ] ) ] [, …n] }
ON <基本表名或视图名> TO <用户名或角色名> [, …n] [ WITH GRANT OPTION ]
```

说明：数据对象的所有者(即创建者)拥有该数据对象上的所有权限，因此，DBA 和对象的所有者有权执行该 GRANT 语句。WITH GRANT OPTION 表示该用户(或角色)还可以向其他用户(或角色)授予所指定的权限。数据对象的操作类型有 SELECT、INSERT、DELETE、UPDATE、REFERENCES。操作类型 ALL 代表上述五种权限。INSERT 和 DELETE 权限会影响整行，因此只可以应用到基本表或视图上，而不能应用到某些列上。SELECT、UPDATE 和 REFERENCES 权限可以有选择性地应用到基本表或视图中的某些列上，如果未指定列，则应用到基本表或视图的所有列上。为创建引用某个表的 FOREIGN KEY 约束，需要在该表上有 REFERENCES 权限。

例 5.11 授予角色 public 在数据库 test2 中查询基本表 C 所有列的权限。

```
GRANT SELECT ON C TO public
```

例 5.12 授予角色 undergraduate 在数据库 test2 中只能查询基本表 T 中的 Tno、Tname、Tsex 和 Title 列的权限。

```
GRANT SELECT(Tno, Tname, Tsex, Title) ON T TO undergraduate
```

例 5.13 授予角色 undergraduate 在数据库 test2 中查询基本表 S 所有列，但只能修改 Mphone 列的权限。

```
GRANT SELECT, UPDATE(Mphone) ON S TO undergraduate
```

例 5.14 授予用户 carluser 在数据库 test2 中查询基本表 SC 所有列的权限，同时允许他将该权限转授给其他用户。

```
GRANT SELECT ON SC TO carluser WITH GRANT OPTION
```

例 5.15 授予用户 carluser 和 maryuser 在数据库 test2 中查询基本表 TC 所有列的权限。

```
GRANT SELECT ON TC TO carluser, maryuser
```

最后要说明的是：一个用户拥有的全部操作权限包括直接授予该用户的权限以及该用户从他所属的各个角色继承得到的权限。

5.3.2 收回权限

已授予的权限可以由 DBA 或其他授权者用 REVOKE 语句收回。

1. 收回模式对象权限

在 T-SQL 语言中，收回模式对象权限的语句格式如下：

```
REVOKE [ GRANT OPTION FOR ] { ALL [ PRIVILEGES ] | <权限> [, …n] }
FROM <用户名或角色名> [, …n] [ CASCADE ]
```

说明：ALL 和权限的含义同授予模式对象权限的 GRANT 语句。GRANT OPTION FOR 表示要收回向其他用户或角色授予指定权限的权限，但不会收回该权限本身。如果用

户或角色具有不带 WITH GRANT OPTION 选项的指定权限，则将收回该权限本身。CASCADE 表示在收回用户或角色该权限的同时，也会收回由其授予其他用户或角色的该权限。

例 5.16 收回用户 carluser 在数据库 test2 中创建基本表和视图的权限。

```
REVOKE CREATE TABLE, CREATE VIEW FROM carluser
```

2. 收回数据对象权限

在 T-SQL 语言中，收回数据对象权限的语句格式如下：

```
REVOKE [ GRANT OPTION FOR ]
{ ALL [ PRIVILEGES ] | <权限> [ ( <列名> [, …n ] ) ] [, …n] }
ON <基本表名或视图名> FROM <用户名或角色名> [, …n] [ CASCADE ]
```

说明：ALL 和权限的含义同授予数据对象权限的 GRANT 语句。GRANT OPTION FOR 表示要收回向其他用户或角色授予指定权限的权限，但不会收回该权限本身。如果用户或角色具有不带 WITH GRANT OPTION 选项的指定权限，则将收回该权限本身。CASCADE 表示在收回用户或角色该权限的同时，也会收回由其授予给其他用户或角色的该权限。

例 5.17 收回角色 undergraduate 在数据库 test2 中查询基本表 S 所有列以及修改 Mphone 列的权限。

```
REVOKE SELECT, UPDATE(Mphone) ON S FROM undergraduate
```

显然执行上述语句后，角色 undergraduate 的成员失去了查询基本表 S 所有列以及修改 Mphone 列的权限。

例 5.18 收回用户 carluser 在数据库 test2 中查询基本表 SC 所有列的权限，同时收回他将该权限转授给其他用户的权限。如果用户 carluser 将查询基本表 SC 所有列的权限授予了其他用户，也一并收回。

首先假定用户 carluser 在获得例 5.14 中的授权后，通过执行 GRANT SELECT ON SC TO maryuser 语句，给用户 maryuser 也授予了查询基本表 SC 所有列的权限。

则 DBA 通过执行 REVOKE SELECT ON SC FROM carluser CASCADE 语句，就完成了例 5.18 中的要求。

如果 DBA 执行的是 REVOKE GRANT OPTION FOR SELECT ON SC FROM carluser CASCADE 语句，则用户 carluser 仍然拥有查询基本表 SC 所有列的权限，但他向其他用户或角色授予该权限的权限被收回。同时 carluser 给用户 maryuser 授予的查询基本表 SC 所有列的权限也被收回。

5.4 SQL Server 的安全机制

为了保证数据库中数据的安全，SQL Server 2014 提供了强大有效的安全管理机制，它能够对用户访问 SQL Server 服务器系统和数据库的安全进行全面的管理。SQL Server 2014 的安全性管理主要包括登录管理、用户管理、角色管理、安全对象与架构以及权限管理

等方面，其中用户管理、角色管理和权限管理在前面论述数据库安全性控制技术时已经作了介绍，而且这些内容与大多数 DBMS 中的相应内容相同或类似。本节对 SQL Server 2014 中的这些内容再做补充介绍。

5.4.1 SQL Server 2014 的身份验证模式

用户登录也就是用户身份验证，是 SQL Server 实施其安全性的第一步。打个比方，SQL Server 是一幢大楼，该大楼内有许多房间，每个房间代表一个数据库，如果用户想对 SQL Server 中的数据库进行操作，首先要做的是进入这幢大楼，用户登录要完成的工作就是进入大楼这个任务。SQL Server 2014 提供两种身份验证模式，一种是 Windows 身份验证，另一种是混合身份验证。

1. Windows 身份验证模式

SQL Server 数据库系统通常运行在 Windows 服务器上，而 Windows 作为网络操作系统本身就具备管理登录、验证账户合法性的能力，因此 Windows 身份验证模式正是利用了 Windows 操作系统本身的安全性和账户管理机制，允许 SQL Server 使用 Windows 的用户名和口令。在这种模式下，用户只需要通过 Windows 的验证，就可以连接到 SQL Server 服务器。Windows 身份验证模式主要有以下优点。

(1) 数据库管理员可以集中管理数据库，而无须管理用户账户。

(2) Windows 有更强的用户账户管理工具，身份验证使用 Kerberos 安全协议，提供强密码复杂性验证，可以设置账户锁定、密码期限等。

(3) Windows 的组策略支持多个用户同时被授权访问 SQL Server。

2. 混合身份验证模式

混合验证模式既允许使用 Windows 身份验证模式，又允许使用 SQL Server 身份验证模式。对于可信用户的连接请求，系统采用 Windows 身份验证模式；而对于非可信用户的连接请求，则采用 SQL Server 身份验证模式。在 SQL Server 身份验证模式下，用户在连接到 SQL Server 服务器时必须提供登录名和登录密码(这些登录信息存储在系统表 syslogins 中，与 Windows 的登录账号无关，在 SQL Server 2014 中系统表 syslogins 对应系统视图 sys.server_principals)，通过验证后，用户就可以连接到 SQL Server 服务器。

SQL Server 身份验证模式主要有以下优点。

(1) 允许 SQL Server 支持那些需要进行 SQL Server 身份验证的旧版应用程序或由第三方提供的应用程序。

(2) 允许 SQL Server 支持具有混合操作系统的环境，在这种环境中并不是所有用户均由 Windows 进行验证。

(3) 允许 SQL Server 支持基于 Web 的应用程序，在这些应用程序中，用户可创建自己的标识。

(4) 允许软件开发人员通过使用基于已知的预设 SQL Server 登录名的复杂权限层次结构来分发应用程序。

但 SQL Server 身份验证模式也存在着不能使用 Kerberos 安全协议、必须在连接时通过网络传递已加密的登录密码、一些自动连接的应用程序将密码存储在客户端，这可能产生安全隐患等缺点。

3. 配置身份验证模式

SQL Server 2014 有两种方法配置身份验证模式。一种是在安装 SQL Server 2014 系统时，会出现选择身份验证模式界面，选择需要的验证模式即可。另一种是对已经安装并选择好验证模式的 SQL Server 2014 服务器，可以修改其验证模式。修改验证模式的操作步骤如下。

首先进入 SQL Server Management Studio 窗口界面，在对象资源管理器中选择服务器，右击，从弹出的快捷菜单中选择“属性”命令；然后在出现的服务器属性窗口界面中选择“安全性”标签，就可以修改身份验证模式。修改完验证模式之后，必须重新启动 SQL Server 2014 服务器才能使设置生效。

5.4.2 SQL Server 2014 的固定角色

角色是一个强大的工具，可以建立一个角色来代表单位中一类员工的工作职责，然后给这个角色授予适当的权限。在根据工作职责定义了一系列角色，并给每个角色指派了适合这项工作的权限之后，就不用管理每个用户的权限，而只需在角色之间移动用户即可。

在 SQL Server 2014 中，角色分服务器角色和数据库角色两种类型，而数据库角色又分为固定数据库角色和用户自定义数据库角色。关于用户自定义数据库角色的管理在 5.2.2 节中已有介绍，下面主要介绍固定服务器角色和固定数据库角色。

1. 固定服务器角色

服务器角色是指被授予管理 SQL Server 服务器权限的角色，它独立于各个数据库，其成员是登录名。服务器角色是内置的(固定的)，即不能添加、修改或删除服务器角色，只能修改角色的成员。SQL Server 2014 共定义了 9 种固定服务器角色，如表 5.1 所示。系统管理员不能将管理服务器的权限直接授予登录名，只有使登录名成为某服务器角色的成员，该登录者才具有该服务器角色的权限。

表 5.1 固定服务器角色

固定服务器角色	描　　述
public	每个登录名均是该角色的成员
sysadmin	系统管理员，拥有所有操作权限，可以执行任何活动
serveradmin	服务器管理员，可以设置服务器范围的配置选项，关闭服务器
setupadmin	安装管理员，可以管理链接服务器和启动过程
securityadmin	安全管理员，可以管理登录和 CREATE DATABASE 权限，还可以读取错误日志和更改密码
processadmin	进程管理员，可以管理在 SQL Server 中运行的进程
dbcreator	数据库创建者，可以创建、更改或删除数据库
diskadmin	磁盘管理员，可以管理磁盘文件
bulkadmin	数据操作管理员，可以执行 BULK INSERT 语句

在 SQL Server 2014 中，可以使用系统存储过程 sp_addsrvrolemember 为某个固定服务器角色添加成员，其格式如下：

```
EXEC sp_addsrvrolemember <登录名> , <角色名>
```

例 5.19 将登录名 carl 添加为固定服务器角色 sysadmin 的成员。

```
EXEC sp_addsrvrolemember  carl , sysadmin
```

可以使用系统存储过程 sp_dropsrvrolemember 将固定服务器角色中的某个成员删除，其格式如下：

```
EXEC sp_dropsrvrolemember <登录名> , <角色名>
```

例 5.20 从固定服务器角色 sysadmin 中删除成员 carl。

```
EXEC sp_dropsrvrolemember  carl , sysadmin
```

2. 固定数据库角色

数据库角色定义在数据库级别上，它可以对数据库进行特定的管理和操作。固定数据库角色是内置的(固定的)，即不能添加、修改或删除固定数据库角色，只能修改角色的成员。每个数据库都有一系列固定数据库角色，尽管在不同的数据库中它们的名称相同，但各个角色的作用域都在各自所属的数据库范围内。SQL Server 2014 共定义了十种固定数据库角色，如表 5.2 所示。

表 5.2 固定数据库角色

固定数据库角色	描　　述
public	每个用户均是该角色的成员
db_owner	在数据库中拥有全部权限
db_accessadmin	可以添加或删除用户
db_securityadmin	可以管理数据库角色和成员，及数据库中的语句和对象权限
db_ddladmin	可以执行任何数据定义语言(DDL)语句
db_backupoperator	可以对数据库进行备份
db_datareader	可以读取数据库内所有用户表中的所有数据
db_datawriter	可以插入、修改或删除数据库内所有用户表中的所有数据
db_denydatareader	拒绝读取数据库内任何用户表中的任何数据
db_denydatawriter	拒绝插入、修改或删除数据库内任何用户表中的任何数据

public 是一个特殊的数据库角色，每个数据库用户都会自动成为 public 角色的成员，并且不能从 public 角色中删除，所以每个数据库用户的默认权限为 public 角色所具有的权限。

5.4.3 拒绝权限

在 SQL Server 2014 中，拒绝权限是指拒绝给当前数据库中的用户或角色授予权限并防止用户或角色通过继承获得权限。收回权限与拒绝权限不同，收回某权限后用户或角色仍然可通过它所属的角色继承来获得该权限，而拒绝某权限后用户或角色将永远无法获得该权限。

1. 拒绝模式对象权限

在 T-SQL 语言中，拒绝模式对象权限的语句格式如下：

```
DENY { ALL [ PRIVILEGES ] | <权限> [, …n] }
TO <用户名或角色名> [, …n] [ CASCADE ]
```

说明：CASCADE 表示在拒绝用户或角色该权限的同时，也会拒绝由其授予给其他用

户或角色的该权限。

例 5.21 在数据库 test2 中，为固定数据库角色 db_ddladmin 添加数据库用户 carluser，并拒绝该用户获得创建基本表和视图的权限。

```
EXEC sp_addrolemember db_ddladmin , carluser
DENY CREATE TABLE, CREATE VIEW TO carluser
```

2. 拒绝数据对象权限

在 T-SQL 语言中，拒绝数据对象权限的语句格式如下：

```
DENY { ALL [ PRIVILEGES ] | <权限> [ ( <列名> [,…n ] ) ] [,…n] }
ON <基本表名或视图名> TO <用户名或角色名> [,…n] [ CASCADE ]
```

说明：CASCADE 表示在拒绝用户或角色该权限的同时，也会拒绝由其授予给其他用户或角色的该权限。

例 5.22 在数据库 test2 中，为固定数据库角色 db_datareader 和 db_datawriter 添加数据库用户 carluser，收回该用户在基本表 S 上查询数据的权限，并拒绝该用户获得基本表 S 上修改和删除数据的权限。

```
EXEC sp_addrolemember db_datareader , carluser
EXEC sp_addrolemember db_datawriter , carluser
REVOKE SELECT ON S FROM carluser
DENY UPDATE, DELETE ON S TO carluser
```

说明：执行上述语句后，carluser 仍然拥有基本表 S 上查询数据的权限，因为他可以通过继承从角色 db_datareader 中获得该权限。但 carluser 无法获得基本表 S 上修改和删除数据的权限，因为已经拒绝他通过继承来获得这些权限。

习 题 5

一、单项选择题

1. TCSEC/TDI 将系统划分为 D CBA 4 组 7 个等级，其中(　　)级是安全产品的最低档次。

A. C1　　B. C2　　C. B1　　D. B2

2. 实现数据库安全性控制的方法和技术有很多，常用的有(　　)。

A. 授权机制　　B. 索引机制　　C. 加锁机制　　D. 建立日志文件

3. 在 SQL Server 中，使用 SSMS 连接数据库引擎时，需要指定登录名和密码，这种安全措施属于(　　)。

A. 授权机制　　B. 视图机制

C. 数据加密　　D. 用户标识与鉴别

4. 在 SQL Server 中，可以使用系统存储过程(　　)为某个数据库角色添加成员。

A. sp_adduser　　B. sp_addrole

C. sp_addrolemember　　D. sp_addsrvrolemember

5. 在 SQL 语言中，用 GRANT 和 REVOKE 语句实现数据库的(　　)。

A. 完整性控制　　B. 安全性控制　　C. 一致性控制　　D. 并发控制

6. 在SQL语言中，可用语句(　　)授予用户U1可以查询数据库DB1中的表T1中的全部数据的权限。

A. GRANT SELECT ON DB1(T1) TO U1

B. GRANT SELECT TO U1 ON DB1(T1)

C. GRANT SELECT ON T1 TO U1

D. GRANT SELECT TO U1 ON T1

7. 在SQL Server中，若希望用户U1具有数据库服务器上的全部权限，则应将其添加到(　　)角色中。

A. sysadmin　　B. serveradmin　　C. db_owner　　D. db_datawriter

8. 在SQL Server中，通常情况下，角色(　　)中的成员不能创建或删除视图。

A. sysadmin　　B. public　　C. db_owner　　D. db_ddladmin

9. 在SQL Server中，可以使用系统存储过程(　　)为某个固定服务器角色添加成员。

A. sp_adduser　　B. sp_addrole

C. sp_addrolemember　　D. sp_addsrvrolemember

10. 在SQL Server中，若要拒绝用户的操作权限，可用语句(　　)。

A. CLEAR　　B. REVOKE　　C. CANCEL　　D. DENY

二、填空题

1. 在网络环境下，数据库的安全体系涉及三个层次：_____层、操作系统层和数据管理系统层。

2. 定义用户权限和_____一起组成了DBMS的安全子系统。

3. 常用的存取控制方法有C2级中的_____存取控制和B1级中的_____存取控制。

4. 安全性控制有许多方法和技术，其中_____间接地实现支持存取谓词的用户权限定义，而_____和_____通常很费时间，DBMS往往都将它们作为可选特征，允许用户根据应用对安全性的要求自由选择使用。

5. 在SQL Server中，每个数据库中都有两个特殊的数据库用户_____和_____，其中_____用户默认是数据库角色db_owner的成员，可以对数据库进行所有操作。

6. 在SQL Server中，每个数据库中都有一个特殊的_____数据库角色，数据库中的每个用户都是该角色的成员。

7. 用户权限由_____和_____两个要素组成，定义用户的存取权限称为授权。

8. SQL Server提供两种身份验证模式，一种是_____身份验证，另一种是_____身份验证。

三、简答题

1. 什么是数据库的安全性？数据库的安全性与数据库的完整性有什么区别和联系？

2. 试述实现数据库安全性控制的常用方法和技术？

3. SQL Server中的登录名和用户有什么区别？如何管理登录名和用户？

4. 试述SQL Server中的角色的含义和作用。

5. SQL Server中如何管理角色？如何管理角色中的成员？

6. 什么是SQL Server中的固定角色？如何分类？如何管理固定角色中的成员？