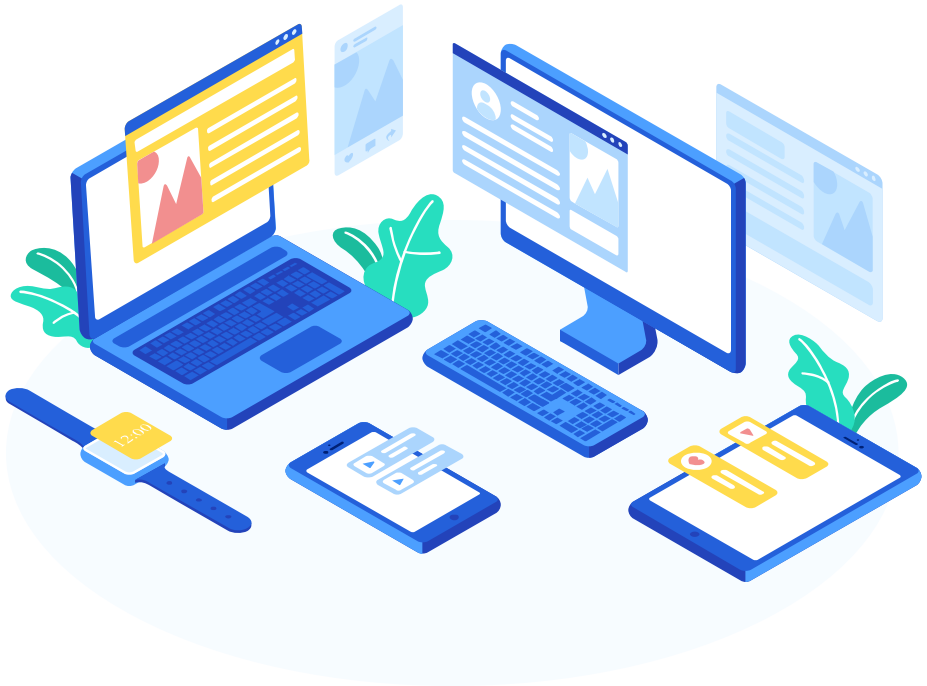
数据库
安全性与完整性

1 数据库安全性

2 数据库完整性

数据库安全性

- 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。
- 数据库系统中大量数据集中存放，且为许多最终用户直接共享，安全性问题更为突出。



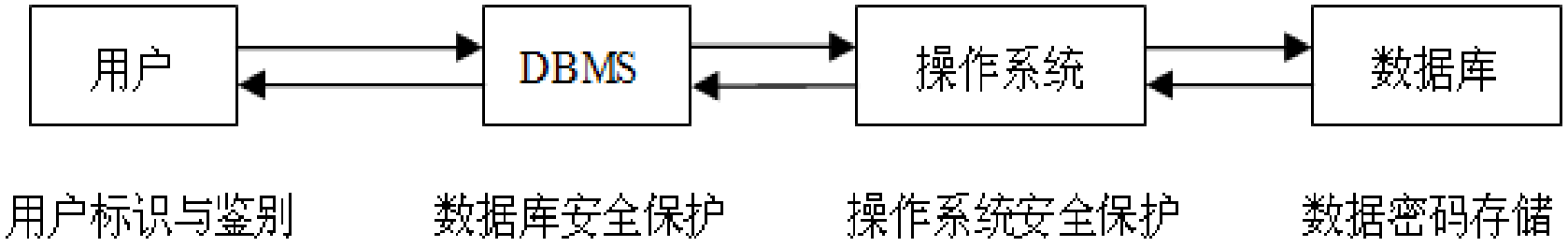
数据库安全性

1. 数据库安全概述

2. SQL Server 安全机制

1. 数据库安全概述

- 数据库安全保护目标是确保只有授权用户才能访问数据库，未被授权的人员则无法接近数据。
- 安全措施是指计算机系统中用户直接或通过应用程序访问数据库所要经过的安全认证过程。
- 数据库安全认证过程如下图所示



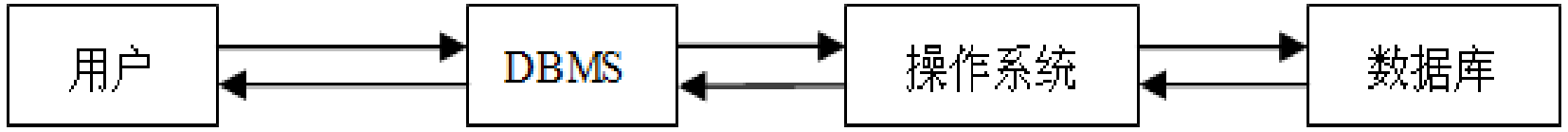
1. 数据库安全概述

■用户标识与鉴别 (identification & authentication)

- 当用户访问数据库时，要先将其用户名 (user name) 与密码 (password) 提交给数据库管理系统进行认证；
- 只有在确定其身份合法后，才能进入数据库进行数据存取操作。

■数据库安全保护

- 通过身份认证的用户，拥有了进入数据库的“凭证”；
- 用户在数据库中执行什么操作，需通过“存取控制”或视图进行权限分配。



三个方面展开说明

1. 数据库安全概述

- **存取控制**：决定用户对数据库中的哪些对象进行操作，进行何种操作。存取控制机制主要包括两部分：

➢定义用户权限及将用户权限登记到数据字典中；

➢合法权限检查，当用户发出操作请求后， DBMS 查找数据字典并根据安全规则进行合法权限检查，若操作请求超出了定义的权限，系统将拒绝执行此操作。

- **视图**：通过为不同的用户定义不同的视图，达到限制用户访问范围的目的。

➢视图机制能隐藏用户无权存取的数据，从而自动地对数据库提供一定程度的安全保护；

➢视图的主要功能在于提供数据库的逻辑独立性，其安全性保护不太精细，往往不能达到应用系统的要求；

➢在实际应用中，通常将视图与存取控制机制结合起来使用，如先通过视图屏蔽一部分保密数据，然后进一步定义存取权限。

1. 数据库安全概述

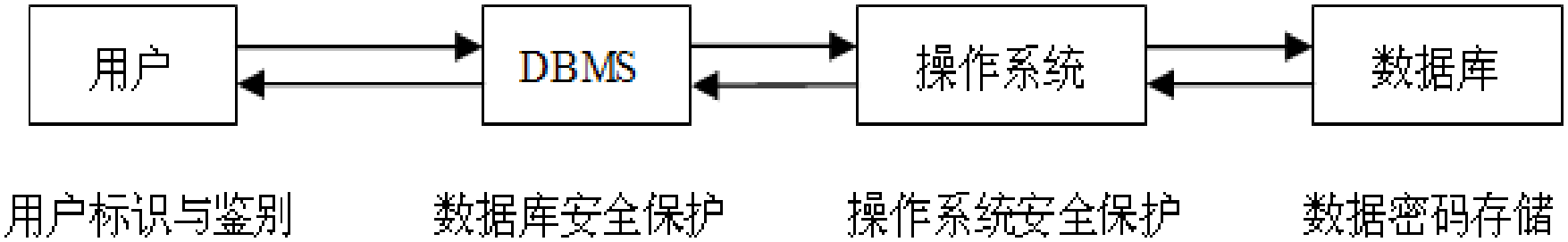
- 审计：是一种监视措施，用于跟踪并记录有关数据的访问活动。
  - 审计追踪把用户对数据库的所有操作自动记录下来，存放在审计日志 (audit log) 中；
  - 审计日志的内容一般包括：
    - ✓ 操作类型 (如修改、查询、删除) ；
    - ✓ 操作终端标识与操作者标识；
    - ✓ 操作日期和时间；
    - ✓ 操作所涉及到的相关数据 (如基本表、视图、记录、属性) ；
    - ✓ 数据库的前映像 (即修改前的值) 和后映像 (即修改后的值) 。
  - 利用这些信息，可找出非法存取数据库的人、时间和内容等；
  - 数据库管理系统往往将审计作为可选特征，允许操作者打开或关闭审计功能。

■操作系统安全保护

- 通过操作系统提供的安全措施来保证数据库的安全性

■数据密码存储

- 访问控制和存取控制可将用户的应用系统访问范围最小化和数据对象操作权限最低化，但对一些敏感数据进行“加密存储”也是系统提供的安全策略；
- 数据加密 (data encryption)： 防止数据库中数据存储和传输失密的有效手段；
- 加密的基本思想： 先根据一定的算法将原始数据 (即明文 , plaintext) 加密为不可直接识别的格式 (即密文 , ciphertext) ， 然后数据以密文的方式存储和传输。



2. SQL Server 安全机制

- 美国、欧洲和国际标准化组织都对计算机系统制定了相应的安全标准。
- 数据库的安全与计算机系统的安全是紧密相关的，数据库系统的安全标准与计算机系统的安全标准是**一致**的。
- 最有影响的标准是美国国防部标准：四组 7 各级别一次为 D,C(C1,C2),B(B1,B2,B3) 和 A(A1) 依次增高，且 C2 中，受控的存取保护通过授权语句 GRANT,REVOKE 完成。

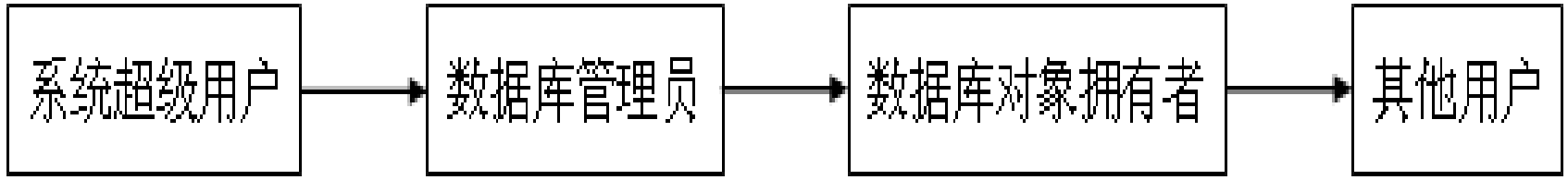
2. SQL Server 安全机制

- SQL 支持受控的存取保护：
  - 在自主存取控制中，用户对不同的数据对象有不同的存取权限；
  - 不同的用户对同一对象有不同的权限；
  - 用户可将其拥有的存取权限转授给其他用户。
- 自主存取控制通过 SQL 的 GRANT 和 REVOKE 语句实现。
- 用户权限：是指用户可以在哪些数据对象上进行哪些类型的操作。它由两个要素组成：数据对象和操作类型。
  - 定义存取权限称为授权 (authorization)；
  - 授权粒度可以精细到字段级，也可以粗到关系级；
  - 授权粒度越细，授权子系统就越灵活，但是系统的开销也会相应地增大。

2. SQL Server 安全机制

■授权分为数据库级、表级和列级权限。

- 在 SQL Server 中权限只能由担任不同角色的用户来分配;
- 不同类型的用户有不同的等级;
- 下图给出了授权等级图。



2. SQL Server 安全机制

- 使用 GRANT 和 REVOKE 语句，向用户授予或收回对数据的操作权限。对数据库模式的授权则由 DBA 在创建用户时实现。
- SQL Server 安全管理机制是架构在认证和权限两大机制下：
  - 认证是指用户必须要有一个登录账号和密码来登录 SQL Server，只有登录后才有访问使用 SQL Server 的入门资格，也只能处理 SQL Server 特定的管理工作；
  - 而数据库内的所有对象的访问权限必须通过权限设置来决定登录者是否拥有某一对象的访问权限。
  - 在数据库内可创建多个用户，然后针对具体对象将对象的创建、读取、修改、插入、删除等权限授予特定的用户。
- 利用角色，SQL Server 管理者可以将某类用户设置为某一角色，这样只对角色进行权限设置便可以实现对该类所有用户权限的设置，大大减少了管理员的工作量。

2. SQL Server 安全机制

■登录账户管理

- 创建登录账号：

[EXECUTE] sp_addlogin [*@loginame=*] '*login*'
[, [*@passwd=*] '*password*'] [, [*@defdb=*] '*database*']

➢ *@loginame='login'* ： *login* 指定登录名称。

➢ *@passwd='password'* ： *password* 指定登录密码，若不指定则默认为 NULL 。

➢ *@defdb='database'* ： *database* 指定登录后用户访问的数据库，若不指定则默认为 master 数据库。

执行系统存储过程 sp_addlogin，除了指定登陆名称之外，其余选项均为可选项。

执行 sp_addlogin 时，必须具有相应的权限。只有 syssdmin 和 securityasmin 固定服务器角色的成员才能执行该命令。

■登录账户管理

- 创建登录账号：

[例 1] 创建登录账号为 login1 、密码为 p666666 的登录账号；
创建登录账号为 login2 、密码为 p888888 的登录账号。

```
sp_addlogin 'login1', 'p666666'
sp_addlogin 'login2', 'p888888'
```

[例 2] 创建登录账号 login3 ， 密码为 p123456 ， 默认的数据库为 ScoreDB 。

```
sp_addlogin login3, 'p123456', 'ScoreDB'
```

此外修改登录账号属性，默认的数据库，删除登录账号等修改。

■用户管理

- 添加用户：

  sp_adduser [*@loginame*=] '*login*'
  [, [*@name_in_db*=] '*user*']

[例 6] 将登录账号 login1 添加到当前数据库 OrderDB 中，且用户名为 u1 。

```
sp_adduser login1, u1
```

[例 7] 将登录账号 login2 添加到当前数据库 OrderDB 中，且用户名为 u2 。

```
sp_adduser login2, u2
```

2. SQL Server 安全机制

■用户管理

- 添加用户：

  sp_adduser [*@loginame*=] '*login*'
  [, [*@name_in_db*=] '*user*']

- 删除用户：

  sp_dropuser [*@name_in_db*=] '*user*'

[例 8] 从当前数据库中删除用户 u1 。

sp_dropuser u1

■权限的授予与收回

- GRANT 和 REVOKE 有两种权限：命令权限和目标权限。
- 命令权限的授予与收回
  - 主要指 DDL 操作权限，语法分别为：

    GRANT {all | <*command_list*>} TO {public | <*username_list*>}

    REVOKE {all | <*command_list*>} FROM {public | <*username_list*>}
  - <*command_list*> 可以是 create database 、 create default 、 create function 、 create procedure 、 create rule 、 create table 、 create view 、 create index 、 backup database 和 backup log 等；
  - 一次可授多种权限，授多种权限时，权限之间用逗号分隔；
  - 如果具有创建对象的 create 权限，则自动具有其创建对象的修改 alter 权限和删除 drop 权限；

2. SQL Server 安全机制

- all： 表示上述所有权限;
- public： 表示所有的用户;
- *<username_list>*： 指定的用户名列表。如果将某组权限同时授予多个用户，则用户名之间用逗号分隔。

[例 9] 将创建基本表和视图的权限授予用户 u1 和 u2：

```
GRANT create table, create view TO u1, u2
```

[例 10] 从用户 u2 收回创建视图的权限：

```
REVOKE create view FROM u2
```

- GRANT和REVOKE有两种权限：命令权限和目标权限。
- 命令权限的授予与收回
  - 主要指DDL操作权限，语法分别为：

```
GRANT {all | <command_list>} TO {public | <username_list>}
REVOKE {all | <command_list>} FROM {public | <username_list>}
```

- **目标权限的授予与收回**
  - 主要指 DML 操作权限，语法分别为：

GRANT {all | <*command_list*>} ON <*objectName*> [(<*columnName_list*>)]
TO {public | <*username_list*>} [WITH GRANT OPTION]

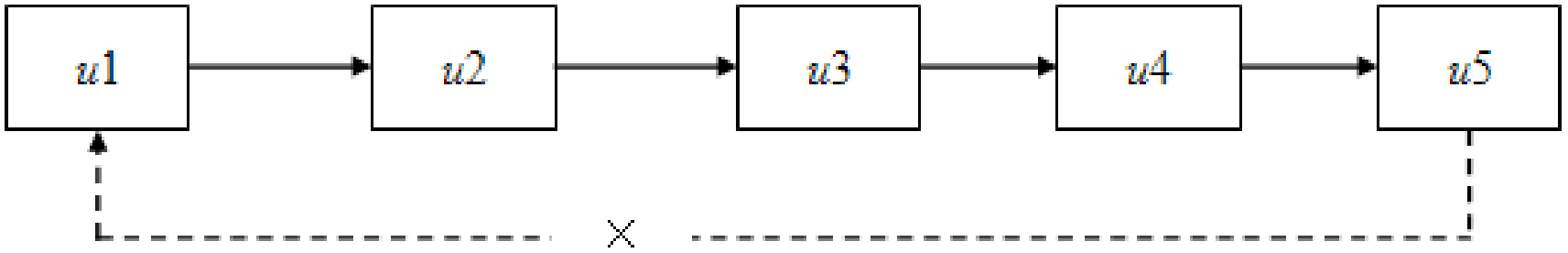
REVOKE {all | <*command_list*>} ON <*objectName*> [(<*columnName_list*>)]
FROM {public | <*username_list*>} [CASCADE | RESTRICT]

  - <*command_list*> 可以是 update 、 select 、 insert 、 delete 、 excute 和 all
    - ✓ excute 针对存储过程授予执行权限；
    - ✓ update 、 select 、 insert 、 delete 针对基本表和视图授权；
    - ✓ all 表示所有的权限。
  - 对象的创建者自动拥有该对象的插入、删除、修改和查询操作权限；存储过程的创建者自动拥有所创建过程的执行权限；

2. SQL Server 安全机制

- CASCADE： 级联收回；
- RESTRICT： 缺省值，若转赋了权限，则不能收回；
- WITH GRANT OPTION： 允许将指定对象上的目标权限授予其它安全用户。

■不允许循环授权，即不允许将得到的权限授予其祖先，如下图所示：



2. SQL Server 安全机制

[例 11] 将存储过程 proSearchBySno 的执行权限授予用户 u1 、 u2 和 u3：

```
GRANT excute ON proSearchBySno TO u1, u2, u3
```

[例 12] 将对班级表 Class 的查询、插入权限授予用户 u1 ， 且用户 u1 可以转授其所获得的权限给其它用户：

```
GRANT select, insert ON Class TO u1 WITH GRANT OPTION
```

[例 13] 将对学生表 Student 的性别、出生日期的查询和修改权限授予用户 u3 、 u4 和 u5 ， 且不可以转授权限：

```
GRANT select, update ON Student(sex, birthday) TO u3, u4, u5
```

- 如果是对列授予权限，命令项可以包括 select **或** update 或两者组合；
- 若使用了 select * ，则必须对表的所有列赋予 select 权限。

2. SQL Server 安全机制

[例 14] 将基本表 Score 的若干权限分别授予用户 u1、u2、u3、u4、u5 和 u6。

- 将表 Score 的所有权限授予用户 u1，且可以转授权限

```
GRANT all ON Score TO u1 WITH GRANT OPTION
```

- 用户 u1 将表 Score 的所有权限授予用户 u2，且可以转授权限

```
GRANT all ON Score TO u2 WITH GRANT OPTION
```

- 用户 u2 将表 Score 的查询和插入权限授予用户 u5，且不可转授权限

```
GRANT select, insert ON Score TO u5
```

- 用户 u2 将表 Score 的所有权限授予用户 u4，且可以转授权限

```
GRANT all ON Score TO u4 WITH GRANT OPTION
```

- 用户 u4 将表 Score 的查询和删除权限授予用户 u6，且可以转授权限

```
GRANT select, delete ON Score TO u6 WITH GRANT OPTION
```

2. SQL Server 安全机制

■ 通过上述的授权操作，用户 u1 、 u2 、 u3 、 u4 、 u5 和 u6 分别得到的权限如下图所示：

授权用户	被授权用户	数据库对象	允许的操作	能否转授权限
DBA	u1, u2, u3	过程 proSearchBySno	执行权限	不能
DBA	u3, u4, u5	基本表 Student	select, update	不能
DBA	u1	基本表 Class	select, insert	能
DBA	u1	基本表 Score	所有权限	能
u1	u2	基本表 Score	所有权限	能
u2	u5	基本表 Score	select, insert	不能
u2	u4	基本表 Score	所有权限	能
u4	u6	基本表 Score	select, delete	能

[例 15] 用户 u2 将转授给用户 u4 的对表 Score 的修改和查询权限收回：

```
REVOKE select, update ON Score FROM u4 CASCADE
```

- 本例必须级联收回，因为用户 u4 将对表 Score 的查询和删除权限转授给了用户 u6 。

[例 16] 用户 u4 将转授给用户 u6 的对表 Score 的查询权限收回：

```
REVOKE select ON Score FROM u6
```

数据库角色

- 数据库角色是指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用户创建一个角色；
- 角色简化了授权操作。

2. SQL Server 安全机制

- 角色的创建、删除、授权、转授和收回语句的语法如下：
  - 角色的创建，使用系统存储过程 sp_addrole 创建角色：

```
sp_addrole [@rolename=] 'role'
```

[例 17] 建立角色 r1 和 r2 。

```
sp_addrole 'r1'
sp_addrole 'r2'
```

2. SQL Server 安全机制

- 角色的创建、删除、授权、转授和收回语句的语法如下：
  - 角色的创建，使用系统存储过程 sp_addrole 创建角色：

    sp_addrole [*@rolename*=] '*role*'

  - 删除数据库角色

    sp_droprole [*@rolename*=] '*role*'

[例 18] 删除数据库角色 r2 。

sp_droprole 'r2'

2. SQL Server 安全机制

- 角色的创建、删除、授权、转授和收回语句的语法如下：
  - 角色的创建，使用系统存储过程 sp_addrole 创建角色：

    sp_addrole [*@rolename=*] '*role*'
  - 删除数据库角色

    sp_droprole [*@rolename=*] '*role*'
  - 增加数据库角色成员

    sp_addrolemember [*@rolename=*] '*role*',
    [*@membername=*] '*security_account*'

[例 19] 将用户 u2 添加到数据库角色 r1 中。

```
sp_addrolemember 'r1', 'u2'
```

2. SQL Server 安全机制

■角色的创建、删除、授权、转授和收回语句的语法如下：

- 角色的创建，使用系统存储过程 sp_addrole 创建角色：

  sp_addrole [*@rolename=*] '*role*'

- 删除数据库角色

  sp_droprole [*@rolename=*] '*role*'

- 增加数据库角色成员

  sp_addrolemember [*@rolename=*] '*role*',
  [*@membername=*] '*security_account*'

- 删除数据库角色成员

  sp_droprolemember [*@rolename=*] '*role*',
  [*@membername=*] '*security_account*'

[例 20] 在数据库角色 r1 中删除用户 u2 。

```
sp_droprolemember 'r1', 'u2'
```

➢只有固定服务器角色 sysadmin 及 db_owner 的成员才能执行该系统存储过程

2. SQL Server 安全机制

- 角色的创建、删除、授权、转授和收回语句的语法如下：
  - 给角色授权：

GRANT {all | *<command_list>*} ON *<objectName>* TO *<role_list>*

  - 将角色授予其他的角色或用户：

GRANT *<role_list>* TO *<role_list>* | *<user_list>*
[WITH ADMIN OPTION]

  - 角色权限的收回：

REVOKE {all | *<command_list>*} ON *<objectName>*
FROM *<role_list>*

  - 从角色或用户中收回角色：

REVOKE *<role>* FROM { *<role_list>* | *<user_list>* }

2. SQL Server 安全机制

■[例 21] 通过角色实现将一组权限授予一个用户。

- 创建一个角色 Role1：

```
sp_addrole 'Role1'
```

- 使用 GRANT 语句，使角色 Role1 拥有 Student 表的 select 、 update 、 insert 权限：

```
GRANT select, update, insert ON Student TO Role1
```

2. SQL Server 安全机制

■[例 21] 通过角色实现将一组权限授予一个用户。

- 创建一个角色 Role1：

```
sp_addrole 'Role1'
```

- 将角色 Role1 授予用户 u1、u2 和 u3，使他们具有角色 Role1 所包含的全部权限：

```
sp_addrolemember 'Role1', 'u1'
sp_addrolemember 'Role1', 'u2'
sp_addrolemember 'Role1', 'u3'
```

或

```
GRANT Role1 TO u1, u2, u3
```

- 通过角色 Role1 可一次性地收回已授予用户 u1 的这 3 个权限：

```
sp_droprolemember 'Role1', 'u1'
```

或

```
REVOKE Role1 FROM u1
```

2. SQL Server 安全机制

■[例 22] 将对基本表 Student 的删除权限授予角色 Role1，并收回查询权限。

```
GRANT delete ON Student TO Role1

REVOKE select ON Student FROM Role1
```

- 通过修改角色的权限，一次性地将用户 u2 和 u3 的权限全部修改了。

3. 触发器实现安全性

■对于复杂的安全性控制，可以通过触发器来实现：

■例如，审计功能、对某数据项限制只有某类用户（如系统管理员）可以修改、限制在某个时间段对数据项可以修改等；

小结

本小节主要讲述了数据库安全性和SQL server 的安全机制等。