

数据库原理

Theory of Database

李静

信息科学与技术学院



问题的提出

- 数据库的一大特点是数据可以共享；
- 数据共享必然带来数据库的安全性问题；
- 数据库系统中的数据共享不能是无条件的共享；

例： 军事秘密、国家机密、新产品实验数据、
市场需求分析、市场营销策略、销售计划、
客户档案、医疗档案、银行储蓄数据



数据库安全性

第11章 安全管理

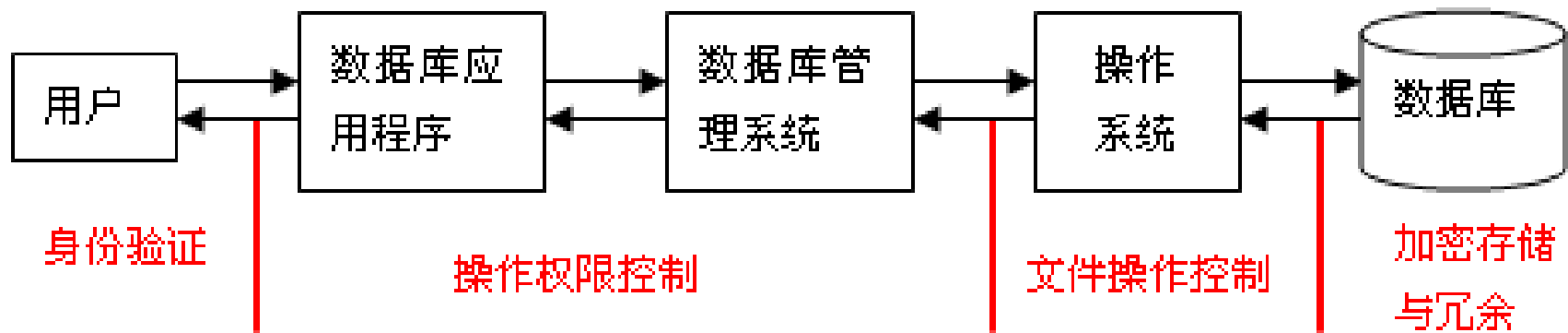
- ❖ 11.1 安全控制概述
- ❖ 11.2 登录名
- ❖ 11.3 数据库用户
- ❖ 11.4 权限管理
- ❖ 11.5 角色



11.1 安全控制概述

- ❖ 数据库的安全控制是指：在数据库应用系统的不同层次提供对有意和无意损害行为的安全防范。
- ❖ 在数据库中
 - 对**有意的非法活动**可采用加密存、取数据的方法控制；
 - 对**有意的非法操作**可使用用户身份验证、限制操作权来控制；
 - 对**无意的损坏**可采用提高系统的可靠性和数据备份等方法来控制。

安全控制模型

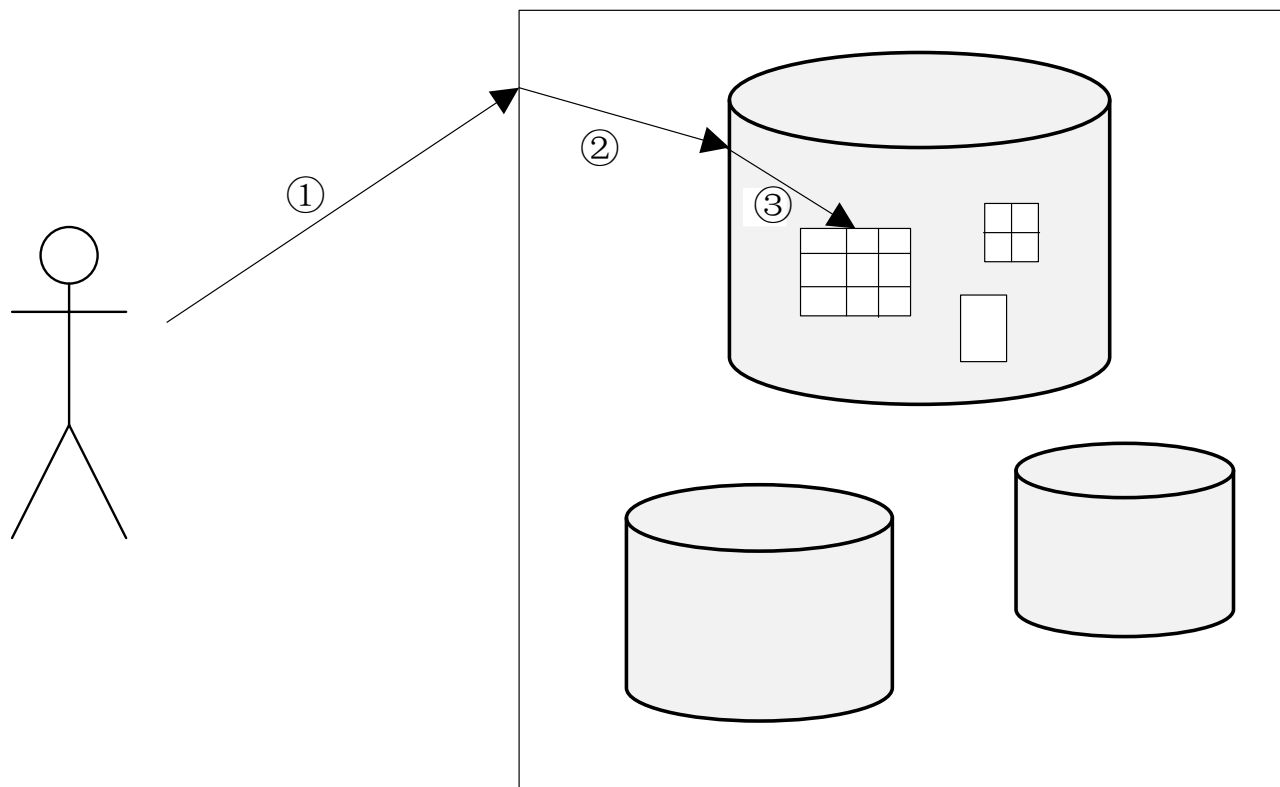


SQL Server 安全控制过程

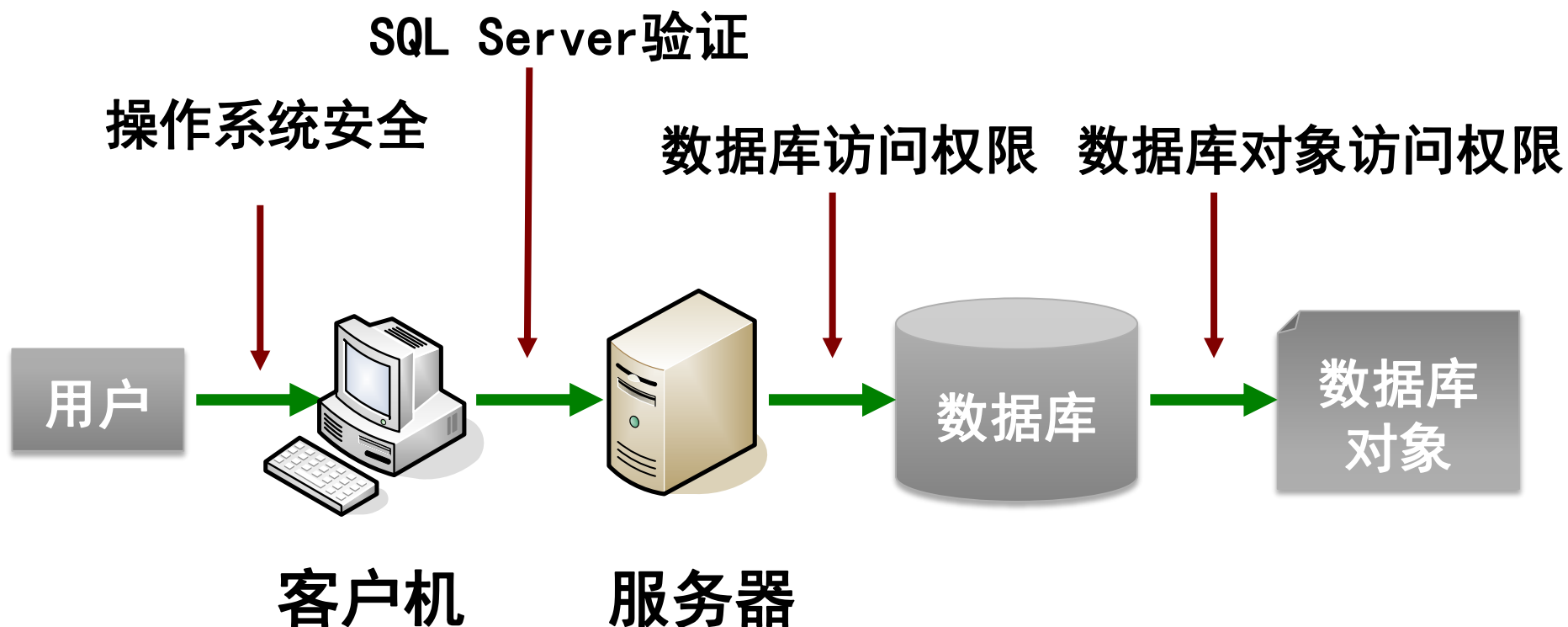
- ❖ 在大型**DBMS**中，用户访问数据库数据要经过三个安全认证过程：
 - 确认用户是否是数据库服务器的合法用户（具有**登录名**）；
 - 第二个过程，确认用户是否是要访问的数据库的合法用户（是**数据库用户**）；
 - 第三过程，确认用户是否具有合适的操作权限（**权限**认证）。

安全认证三个过程示意图

数据库服务器



SQL Server的安全认证模式



SQL SERVER安全体系结构

登录名

- ❖ **SQL Server** 的安全权限是基于标识用户身份的登录标识符（**Login ID**，登录ID），登录ID就是控制访问**SQL Server** 数据库服务器的登录名。
- ❖ **11.2.1 身份验证模式**
- ❖ **11.2.2 建立登录名**
- ❖ **11.2.3 删除登录名**

身份验证模式

❖ SQL Server 支持两类登录名：

- 由SQL Server负责验证的登录名；
- Windows网络账户，可以是组用户。

❖ SQL Server 相应地提供了两种身份验证模式：

- Windows身份验证模式
- 混合验证模式

Windows身份验证模式

- ❖ **SQL Server**将用户的身份验证交给了 **Windows** 操作系统来完成。
- ❖ 在这种身份验证模式下，**SQL Server** 将通过 **Windows**操作系统来获得用户信息，并对登录名和密码进行重新验证。
- ❖ 使用**Windows**身份验证模式时，用户必须先登录到 **Windows** 操作系统，然后再登录到 **SQL Server**。

混合身份验证模式

- ❖ 允许**Windows**授权用户和**SQL**授权用户登录到**SQL Server**数据库服务器。
- ❖ 如果希望允许非**Windows**操作系统的用户也能登录到**SQL Server**数据库服务器上，则应该选择混合身份验证模式。
- ❖ **SQL Server**身份验证的登录信息保存在**SQL Server**实例上；**Windows**身份验证的登录信息由**Windows**和**SQL Server**实例共同保存。

设置身份验证模式

- ❖ 系统管理员根据系统的实际应用情况设置**SQL Server**的身份验证模式。
- ❖ 可以在安装**SQL Server** 时设置
- ❖ 也可以在安装完成后通过**SSMS**工具进行设置。

设置方法

- ❖ 在**SSMS**的对象资源管理器中，在**SQL Server**实例上右击鼠标，选择“属性”命令。
- ❖ 在“服务器属性”窗口左边的“选择页”上，单击“安全性”选项。
- ❖ 在“服务器身份验证”部分，可以设置该实例的身份验证模式。

SQL Server的安全认证模式

认证模式

身份验证阶段

身份验证通过，表示用户可以连接到SQL Server 服务器上。

权限认证阶段

检测用户是否有访问服务器数据的权限，为此需要授予每个数据库中映射到用户登录的账户访问权限，权限认证可以控制用户对数据库进行操作。

身份验证和权限认证

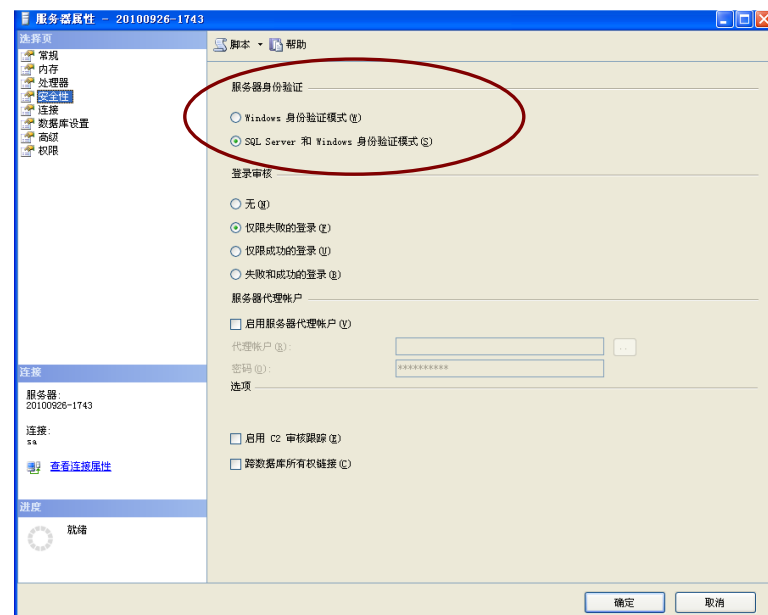
Windows身份验证模式

使用Windows操作系统的安全机制来验证用户身份。只要用户能够通过Windows用户身份验证，即可连接到SQL Server 服务器上。

混合身份验证模式

使用Windows身份验证或SQL Server身份验证与SQL Server服务器连接。

➔ 设置身份验证模式



创建登录账号

登录账号

登录服务器的登录
账号

用户账号

数据库用户账号

The screenshot shows the '连接到服务器' (Connect to Server) dialog box. The title bar says '连接到服务器'. The main area has a header for 'Microsoft SQL Server 2005' and 'Microsoft Windows Server System'. The fields are as follows:

- 服务器类型 (T): 数据库引擎
- 服务器名称 (S): 20100926-1743
- 身份验证 (A): SQL Server 身份验证
- 登录名 (L): sa (This field is circled in red)
- 密码 (P): [Empty]
- ☐ 记住密码 (M)

At the bottom, there are buttons for '连接 (C)', '取消', '帮助', and '选项 (O) >>'.

举例：一坐大楼是一个数据库管理系统，每个房间代表一个数据库，房间里的资料可以表示数据库对象（表、视图等）。则**登录名就相当于进入大楼的钥匙**，而**用户名相当于进入每个房间的钥匙**。房间中的资料是根据用户名的不同而有不同权限的。

建立登录名

❖ 建立登录名有两种方法：

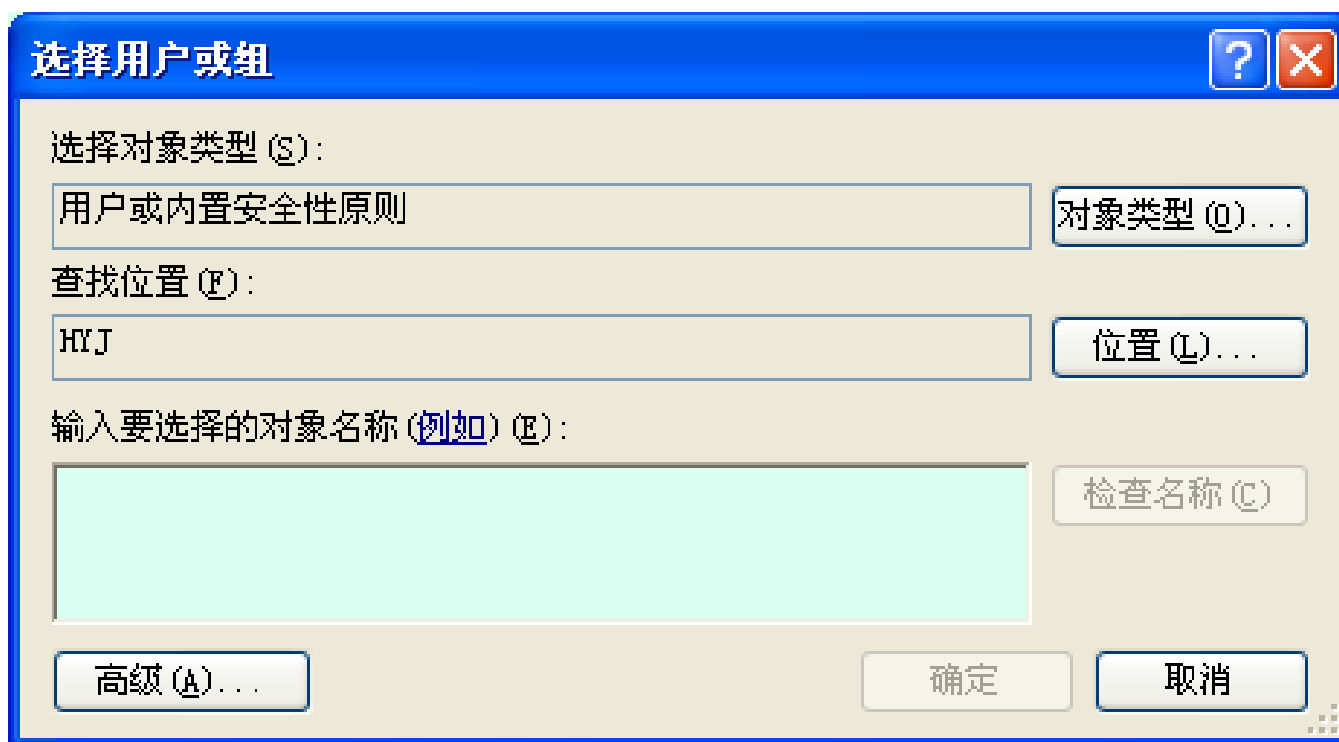
- 通过**SSMS**工具实现，
- 通过**T-SQL**语句实现。



建立Windows身份验证的登录名

- ❖ 使用**Windows**登录名进行的连接，被称为信任连接。
- ❖ 在**SSMS**的对象资源管理器中，依次展开“安全性” → “登录名”节点。
- ❖ 在“登录名”节点上右击鼠标，选择“新建登录名”命令。























❖ 单击“搜索”按钮，弹出“选择用户或组”窗口。



❖ 单击“高级”按钮，弹出“选择用户或组”窗口。



❖ 单击“立即查找”按钮，在下面将列出查找的结果。

名称 (RDN)	在文件夹中
 Administrator	HYJ
 ANONYMOUS LOGON	
 Authenticated Users	
 BATCH	
 CREATOR GROUP	
 CREATOR OWNER	
 DIALUP	
 Everyone	
 gly	HYJ
 Guest	HYJ
 HelpAssistant	HYJ
 INTERACTIVE	
 LOCAL SERVICE	
 NETWORK	
 NETWORK SERVICE	
 REMOTE INTERACTIVE LOGON	
 SERVICE	
 SUPPORT_388945a0	HYJ
 SYSTEM	
 TERMINAL SERVER USER	
 Win_User1	HYJ
 Win_User2	HYJ

建立SQL Server身份验证的登录名

- ❖ 在**SSMS**的对象资源管理器中，依次展开“安全性” → “登录名”节点。
- ❖ 在“登录名”节点上右击鼠标，选择“新建登录名”命令。
- ❖ 在弹出的窗口中输入登录名。

登录名 - 新建

选择页

- 常规
- 服务器角色
- 用户映射
- 安全对象
- 状态

连接

服务器: HYJ

连接: HYJ\gly

[查看连接属性](#)

进度

就绪

脚本 帮助

登录名 (N): SQL_Meer1

☐ Windows 身份验证 (W)

☒ SQL Server 身份验证 (S)

密码 (P):

确认密码 (C):

☒ 强制实施密码策略 (F)

☒ 强制密码过期 (X)

☒ 用户在下次登录时必须更改密码 (U)

☐ 映射到证书

证书名称 (T):

☐ 映射到非对称密钥

密钥名称 (K):

默认数据库 (D): master

默认语言 (A): <默认值>

确定 取消

登录名必须已存在
操作系统的登录账号中

选中的话，必须按照一定的密码策略来设置；不选，可设置任意位数。

如果选中，可以使用密码过期策略来校验密码。

新建登录窗口的一些选项说明

❖ 强制密码过期

- 对该登录名强制实施密码过期策略。必须先选中“强制实施密码策略”才能启用此复选框。

❖ 用户在下次登录时必须更改密码

- 首次使用新登录名时，**SQL Server** 将提示用户输入新密码。

新建登录窗口的一些选项说明

❖ 默认数据库

- 指定该登录名初始登录到**SSMS**时进入的数据库。

❖ 默认语言

- 指定该登录名登录到**SQL Server**时使用的默认语言。

用T-SQL语句建立登录名

```
CREATE LOGIN login_name  
{ WITH <option_list1> | FROM <sources> }  
  
<sources> ::=  
WINDOWS [WITH <windows_options> [...]  
]  
  
<option_list1> ::=  
PASSWORD = ' password '  
[ , <option_list2> [ ,... ] ]
```

建立登录帐户的T-SQL语句（续）

<option_list2> ::=

SID = sid

| DEFAULT_DATABASE = database

| DEFAULT_LANGUAGE = language

<windows_options> ::=

DEFAULT_DATABASE = database

| DEFAULT_LANGUAGE = language

示例

- ❖ 例1. 创建SQL Server身份验证的登录帐户。登录名为：SQL_User2，密码为：a1b2c3XY。

```
CREATE LOGIN SQL_User2  
WITH PASSWORD =  
'a1b2c3XY'
```

示例

❖ 例2. 创建Windows身份验证的登录帐户。从Windows域帐户创建 [HYJ\Win_User2] 登录帐户。

```
CREATE LOGIN [HYJ\Win_User2]  
FROM WINDOWS
```

示例

❖ 例3. 创建SQL Server身份验证的登录帐户。

登录名为: **SQL_User3**, 密码为:

AD4h9fcdhx32MOP。要求该登录帐户首次连接服务器时必须更改密码。

```
CREATE LOGIN SQL_User3
```

```
WITH PASSWORD = 'AD4h9fcdhx32MOP'
```

```
MUST_CHANGE
```

11.2.3 删除登录名

- ❖ 依次展开“安全性” → “登录名”节点。
- ❖ 在要删除的登录名上右击鼠标，选择“删除”命令，弹出删除登录名属性窗口。
- ❖ 单击“确定”按钮。

删除登录帐户的T-SQL语句

DROP LOGIN login_name

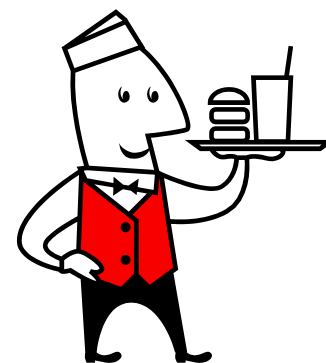
- ❖ 不能删除正在使用的登录帐户，也不能删除拥有任何数据库对象、服务器级别对象的登录帐户。
- ❖ 例1. 删除**SQL_User2**登录帐户。

DROP LOGIN SQL_User2

11.3 数据库用户

❖ 11.3.1 建立数据库用户

❖ 11.3.2 删除数据库用户



11.3.1 建立数据库用户

- ❖ 展开要建立数据库用户的数据库。
- ❖ 展开“安全性”节点，在“用户”节点上右击鼠标，选择“新建用户”命令。
- ❖ 在“登录名”部分指定将要成为此数据库用户的登录名。单击“登录名”文本框右边的 按钮，可以查找某登录名。

用T-SQL语句建立数据库用户

CREATE USER user_name [{ { **FOR** | **FROM** }
{ **LOGIN** login_name }]

- ❖ **user_name**: 指定在此数据库中用于识别该用户的名称。
- ❖ **LOGIN login_name**: 指定要映射为数据库用户的SQL Server登录名。
- ❖ 如果省略**FOR LOGIN**, 则新的数据库用户将被映射到同名的SQL Server登录名。

示例

- ❖ 让**SQL_User2**登录帐户成为当前数据库中的用户，并且用户名同登录名。

CREATE USER SQL_User2

示例

- ❖ 本示例首先创建名为**SQL_JWC**且具有密码的**SQL Server**身份验证的服务器登录名，然后在**students**数据库中创建与此登录帐户对应的数据库用户**JWC**。

```
CREATE LOGIN SQL_JWC  
WITH PASSWORD = 'jKJI3$nN09jsK84';  
GO  
USE students;  
GO  
CREATE USER JWC FOR LOGIN  
SQL_JWC;
```

注意

- ❖ 服务器登录名与数据库用户是两个完全不同的概念。
 - 具有登录名的用户可以登录到**SQL Server**实例上，而且只局限在实例上进行操作。
 - 数据库用户则是登录名以什么样的身份在该数据库中进行操作，是登录名在具体数据库中的映射，这个**映射名**（数据库用户名）可以和登录名一样，也可以不一样

11.3.2 删除数据库用户

- ❖ 删除数据库用户，实际就是解除了登录名和数据库用户之间的映射关系。
- ❖ 删除数据库用户之后，其对应的登录名仍然存在。
- ❖ 删除方法：
 - 展开“数据库” → “students” → “安全性” → “用户”节点。
 - 在要删除的用户名上右击鼠标，选择“删除”命令。

用T-SQL语句删除数据库用户

❖ 语句

DROP USER user_name

❖ 其中**user_name**为要在此数据库中删除的用户名。

❖ 示例. 删除**SQL_User2**用户。

DROP USER SQL_User2

11.4 管理权限

❖ 11.4.1 权限种类及用户分类

❖ 11.4.2 权限的管理



权限种类

❖ 对象权限

- 是对表、视图等对象中数据的操作权。

❖ 语句权限

- 创建对象的权限。

❖ 隐含权限

- 指由**SQL Server**预定义的服务器角色、数据库角色、数据库拥有者和数据库对象拥有者所具有的权限。

数据库用户的分类

❖ 系统管理员

- 在数据库服务器上具有全部的权限。
- **SQL Server 2005**的默认系统管理员是“**sa**”。

❖ 数据库对象所有者

- 创建数据库对象的用户即为数据库对象所有者。
- 数据库对象所有者对其所拥有的对象具有全部权限。

❖ 普通用户

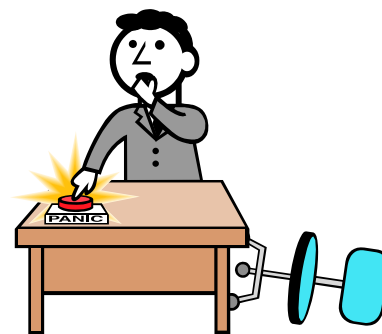
- 只具有对数据库数据的增、删、改、查权限。

11.4.2 权限的管理

- ❖ **授予权限**：允许用户或角色具有某种操作权
- ❖ **收回权限**：不允许用户或角色具有某种操作权，或者收回曾经授予的权限。
- ❖ **拒绝权限**：拒绝某用户或角色具有某种操作权，既使用户或角色由于继承而获得这种操作权，也不允许执行相应的操作。

对象权限的管理

- ❖ 可以通过**SSMS**工具实现，
- ❖ 也可以通过**T-SQL**语句实现。

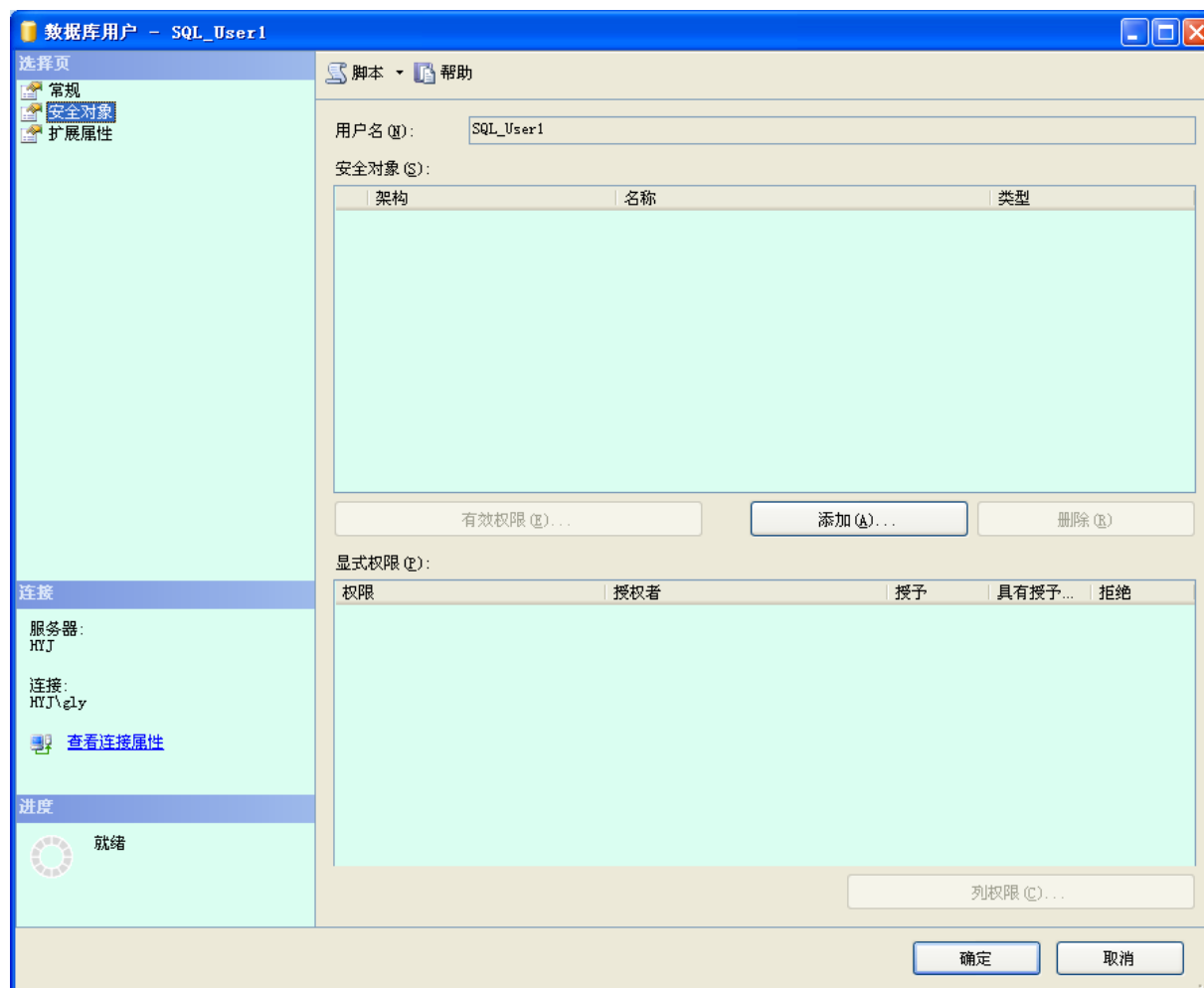


用SSMS工具实现

- ❖ 展开某数据库下的“安全性” → “用户”，
- ❖ 在要授权的用户上右击鼠标，在弹出的菜单中选择“属性”命令，弹出数据库用户属性窗口。
- ❖ 单击窗口左边“选择页”中的“安全对象”选项，出现“安全对象”窗口。

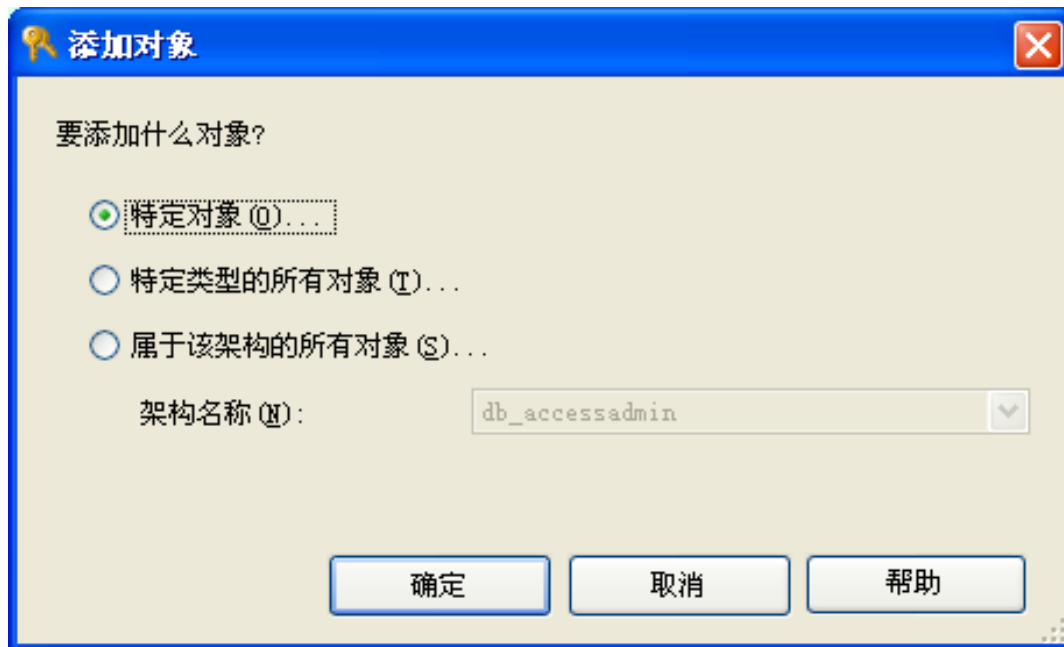
数据库用户属性中的“安全对象”页

单击“添加”
弹出“添加
对象”窗口



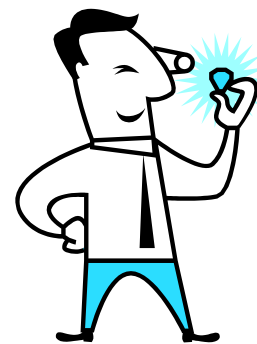
“添加对象”窗口

- ❖ 在这个窗口中可以选择要添加的对象类型。
- ❖ 默认是添加“特定对象”类



用T-SQL语句实现权限管理

- ❖ 用于管理权限的**T-SQL**语句有三个：
- ❖ **GRANT**：用于授予权限。
- ❖ **REVOKE**：用于收回或撤消权限。
- ❖ **DENY**：用于拒绝权限



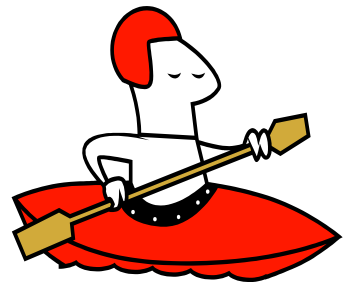
授权语句

❖ **GRANT** 对象权限名 [, ...]

ON { 表名 | 视图名 }

TO {数据库用户名| 用户角色名 }

[, ...]



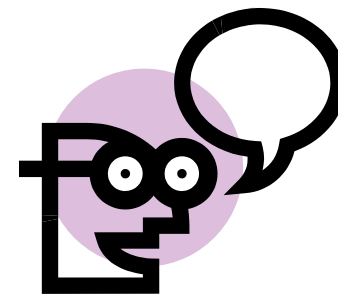
收权语句

REVOKE 对象权限名 [, ...]

ON { 表名|视图名 }

FROM { 数据库用户名|用户角色名 }

[, ...]



拒绝语句

DENY 对象权限名 [, ...]

ON {表名|视图名 }

TO {数据库用户名|用户角色名 }

[, ...]



示例

❖ 例1. 为用户user1授予Student表的查询权。

GRANT SELECT ON Student TO user1

❖ 例2. 为用户user1授予SC表的查询权和插入权。

GRANT SELECT, INSERT ON SC TO user1

❖ 例3. 收回用户user1对Student表的查询权。

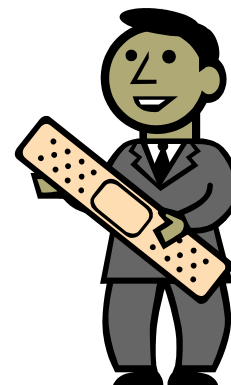
REVOKE SELECT ON Student FROM user1

❖ 例4. 拒绝user1用户具有SC表的更改权。

DENY UPDATE ON SC TO user1

语句权限管理

- ❖ 用**SSMS**实现：基本同对象权限管理。
- ❖ 用**T-SQL**语句实现。



语句权限管理的T-SQL语句

❖ 授权语句

GRANT 语句权限名 [, ...]

TO {数据库用户名 | 用户角色名}

[, ...]

语句权限管理

❖ 收权语句

REVOKE 语句权限名 [, ...]

FROM { 数据库用户名 | 用户角色名 }

[, ...]

收权语句

❖ 拒绝语句

DENY 语句权限名 [, ...]

TO {数据库用户名 | 用户角色名}

[, ...]

示例

- ❖ 例5. 授予user1具有创建表的权限。

GRANT CREATE TABLE TO user1

- ❖ 例6. 授予user1和user2具有创建表和视图的权限。

**GRANT CREATE TABLE, CREATE VIEW TO
user1, user2**

- ❖ 例7. 收回授予user1创建表的权限。

REVOKE CREATE TABLE FROM user1

- ❖ 例8. 拒绝user1创建视图的权限。

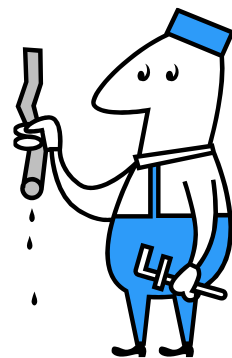
DENY CREATE VIEW TO user1

11.5 角色

❖ 为便于对用户及权限的管理，可以将一组具有相同权限的用户组织在一起，这一组具有相同权限的用户就称为**角色(Role)**。

❖ **SQL Server** 中，角色分为：

- 固定的服务器角色
- 固定的数据库角色
- 用户自定义的角色——介绍



11.5.1 建立用户定义的角色

- ❖ 属于数据库一级的角色。
- ❖ 用户可以根据实际的工作职能情况定义自己的一系列角色，并给每个角色授予合适的权限。
- ❖ 用户自定义的角色的成员可以是数据库的用户，也可以是用户定义的角色。

用SSMS建立用户角色

- ❖ 展开某数据库下的“安全性” → “角色”节点，
- ❖ 在“角色”上右击鼠标，在弹出的菜单中选择“新建”下的“新建数据库角色”命令，或者是在“角色”节点下的“数据角色”上右击鼠标，在弹出的菜单中选择“新建数据库角色”命令，
- ❖ 在弹出“新建数据库角色”窗口中进行相应设置。

用T-SQL语句建立用户定义的角色

CREATE ROLE role_name

[AUTHORIZATION owner_name]

- ❖ **role_name**: 待创建角色的名称。
- ❖ **AUTHORIZATION owner_name**: 将拥有新角色的数据库用户或角色。如果未指定用户，则执行**CREATE ROLE**的用户将拥有该角色。

示例

例1.创建用户自定义角色：**CompDept**，拥有者为创建该角色的用户。

```
CREATE ROLE CompDept;
```

例2.创建用户自定义角色：**InfoDept**，拥有者为**SQL_User1**。

```
CREATE ROLE InfoDept  
AUTHORIZATION SQL_User1;
```


11.5.2 为用户定义的角色授权

- ❖ 对用户角色的授权的方法与为数据库用户授权方法完全一样。
- ❖ 例3.为**Software**角色授予**students**数据库中**Student**表的查询权。

GRANT SELECT ON Student TO Software

- ❖ 例4.为**Admin**角色授予**students**数据库中**Student**表的增、删、改、查权。

**GRANT SELECT,INSERT,DELETE,UPDATE
ON Student TO Admin**

为用户定义的角色添加成员

- ❖ 角色中的成员自动具有角色的全部权限，因此在为角色授权之后，就需要为角色添加成员了。
- ❖ 为角色添加成员可以用**SSMS**工具实现，也可以同**T-SQL**语句实现。

用SSMS工具实现

- ❖ 展开某数据库下的“安全性” → “角色”节点，
- ❖ 在要添加成员的角色上右击鼠标，在弹出的菜单中选择“属性”命令，弹出数据库角色属性窗口。
- ❖ 单击“添加”按钮，弹出“选择数据库用户或角色”窗口。
- ❖ 进行相应选择...

用T-SQL语句实现

- ❖ 使用**sp_addrolemember**系统存储过程（存储过程是一段可调用执行的代码）：

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

- ❖ **[@rolename =] 'role'**：当前数据库中的数据库角色的名称。
- ❖ **[@membername =] 'security_account'**：要添加到角色中的数据库用户名。
security_account可以是数据库用户、数据库角色、Windows 登录名或Windows组。

示例

- ❖ 例5. 将Windows登录名HYJ\Win_User1添加到当前数据库的Software角色中。

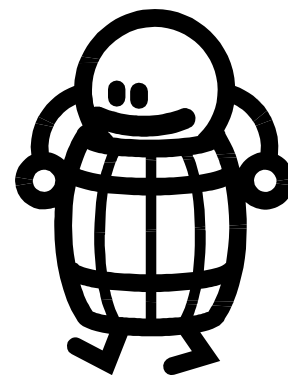
```
EXEC sp_addrolemember 'Software',  
    'HYJ\Win_User1'
```

- ❖ 例6. 将SQL_User2添加到当前数据库的Admin角色中（假设该角色已存在）。

```
EXEC sp_addrolemember 'Admin',  
    'SQL_User2'
```

删除用户定义角色中的成员

- ❖ 当不希望某用户是某角色中的成员时，可将用户从角色中删除。
- ❖ 从角色中删除成员可以通过**SSMS**工具实现，也可以通过**T-SQL**语句实现。



用SSMS工具实现

- ❖ 展开某数据库下的“安全性” → “角色”节点，
- ❖ 在要添加成员的角色上右击鼠标，在弹出的菜单中选择“属性”命令，弹出数据库角色属性窗口。
- ❖ 在窗口中，选中要删除的成员名，单击“删除”按钮即可从角色中删除将所选成员。


用T-SQL语句实现

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

- ❖ **[@rolename =] 'role'**：将从中删除成员的数据库角色名。
- ❖ **[@membername =] 'security_account'**：被从数据库角色中删除的用户名。
- ❖ **例7. 删除Admin角色中的SQL_User2成员。**

**EXEC sp_droprolemember
'Admin','SQL_User2'**

补充：视图机制

 把要保密的数据对无权存取这些数据的用户隐藏起来，对数据提供一定程度的安全保护。

➡ 在一定程度上提供了数据独立性；

➡ 间接实现了支持存取谓词的用户权限定义。

补充：视图机制

[例]建立计算机系学生的视图，把对该视图的SELECT权限授予王平，把该视图上的所有操作权限授予张明。



先建立计算机系学生的视图
CS_Student

```
CREATE VIEW CS_Student
AS
SELECT *
FROM Student
WHERE Sdept='CS';
```

补充：视图机制

[例]建立计算机系学生的视图，把对该视图的SELECT权限授予王平，把该视图上的所有操作权限授予张明。



在视图上进一步定义存取权限

```
GRANT SELECT
    ON CS_Student
    TO 王平 ;

GRANT ALL PRIVILIGES
    ON CS_Student
    TO 张明;
```

补充：审计

什么是审计?

➡ 审计日志 (Audit Log)

✓ 将用户对数据库的所有操作记录在上面

➡ DBA利用审计日志

✓ 找出非法存取数据的人、时间和内容

➡ C2以上安全级别的DBMS必须具有审计功能

补充：审计

审计

用户级审计

针对自己创建的数据库表或视图进行审计；

记录所有用户对这些表或视图的一切成功
和（或）不成功的访问要求以及各种类型
的SQL操作；

系统级审计

由DBA设置；

监测成功或失败的登录要求；

监测GRANT和REVOKE操作以及其他
数据库级权限下的操作。

补充：审计



AUDIT语句：设置审计功能；



NOAUDIT语句：取消审计功能；

〔例〕对修改SC表结构或修改SC表数据的操作进行审计。

```
AUDIT ALTER, UPDATE  
ON SC;
```

〔例〕取消对SC表的一切审计。

```
NOAUDIT ALTER, UPDATE  
ON SC;
```

补充：数据加密

数据加密

防止数据库中数据在存储和传输中失密的有效手段。

加密的基本思想：

根据一定的算法将原始数据（明文）变换为不可直接识别的格式（密文），从而使得不知道解密算法的人无法获知数据内容。

加密方法

- 替换方法：使用密钥用密文替换明文内容
- 置换方法：排列顺序
- 混合方法

DBMS中的数据加密

补充：统计数据库安全性



统计数据库

- ✧ 允许用户查询聚集类型的信息（如合计、平均值等）；
- ✧ 不允许查询单个记录信息；



统计数据库中特殊的安全性问题

- ♣ 隐蔽的信息通道；
- ♣ 能从合法的查询中推导出不合法的信息；

补充：统计数据库安全性

三大规则

- 1: 任何查询至少要涉及 N (N 足够大)个以上的记录;
- 2: 任意两个查询的相交数据项不能超过 M 个;
- 3: 任一用户的查询次数不能超过 $1+(N-2)/M$;

补充：统计数据库安全性

数据库安全机制的设计目标

试图破坏安全的人所花费的代价 >> 得到的利益

小结

1 安全性标准：TCSEC和CC

2 实现数据库系统安全性的技术和方法

- ◆ 存取控制技术
- ◆ 视图技术
- ◆ 审计技术

3 自主存取控制功能

- ◆ 通过SQL 的GRANT语句和REVOKE语句实现

4 角色

- ◆ 使用角色来管理数据库权限可以简化授权过程
- ◆ CREATE ROLE语句创建角色
- ◆ GRANT 语句给角色授权

作业



P172

Theory of Database