1. 计算机安全性概论
2. 数据库安全性控制
3. 非法使用数据库的情况

编写合法程序绕过数据库管理系统及其授权机制

直接或编写应用程序执行非授权操作

通过多次合法查询数据库从中推导出一些保密数据

1. 安全措施

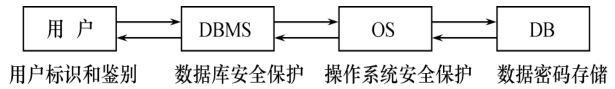
计算机系统中，安全措施是一级一级层层设置

系统根据用户标识鉴定用户身份，合法用户才准许进入计算机系统

数据库管理系统还要进行存取控制，只允许用户执行合法操作

操作系统有自己的保护措施

数据以密码形式存储到数据库中



1. 用户身份鉴别
2. 用户身份鉴别

系统提供的最外层安全保护措施

用户标识：由用户名和用户标识号组成（用户标识号在系统整个生命周期内唯一）

1. 用户身份鉴别的方法

静态口令鉴别：静态口令一般由用户自己设定，这些口令是静态不变的

动态口令鉴别：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登录数据库管理系统，即采用一次一密的方法（如手机短信验证码）

生物特征鉴别：通过生物特征进行认证的技术，生物特征如指纹、虹膜和掌纹等

智能卡鉴别：智能卡是一种不可复制的硬件，内置集成电路的芯片，具有硬件加密功能

1. SQL Server 2008安全验证模式

当用户使用SQL SERVER2008时，需要经过两个安全性阶段，**身份验证**和**权限认证**阶段

·身份验证阶段:用户在SQL SERVER2008上获得任何数据库访问权限之前，必须首先登录到SQLSERVER2008并且是合法的，否则服务器将拒绝用户登录

·权限验证阶段:身份验证阶段只能验证用户**是否具有连接到SQL SERVER2008的权限**，通过身份验证后，需要验证用户**是否具有访问服务器数据的权限**，为此需要为每个数据库建立用户，并将账户映射到登录账户，并为用户分配对象的访问权限

·SQL Server 2008提供了两种确认用户对**数据库引擎服务**的验证模式：

Windows身份验证:允许SQL Server可以使用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，登录SQLServer时就不再需要输入帐户和密码了

·SQL Server身份验证:SQL Server身份验证模式要求用户在连接SQL Server时必须提供登录名和登录密码，与Windows的登录帐号无关。SQL Server自身执行认证处理。 利用这种方式可以很方便地从网络上访问sql server服务器

·创建登录名

CREATE LOGIN Zhangsan

WITH PASSWORD = ‘abc123!’

每个用来登录数据库的帐户都是一个用户。**通过用户这个对象，可以设置数据库的使用权限**。同一个数据库可以拥有多个用户，同一个用户也可以同时访问多个数据库（**用户与数据库是多对多的**）

创建登录名后，不能切换db，但可以看到有什么db

·创建用户

CREATE USER dbUser1 FOR LOGIN zhangsan（执行此语句时，正在使用的那个db就是这个用户所属于的db，db与用户是多对多的）

WITH DEFAULT\_SCHEMA = student;//默认使用哪个schema

给定user后，可切换到所属db，但看不到表，不可切换别的db

1. 存取控制

存取控制机制组成：定义用户权限、合法权限检查

用户权限定义和合法权检查机制一起组成了DBMS的安全子系统

自主存取控制（Discretionary Access Control ，简称DAC）：C2级，灵活

强制存取控制（Mandatory Access Control，简称 MAC）：B1级，严格

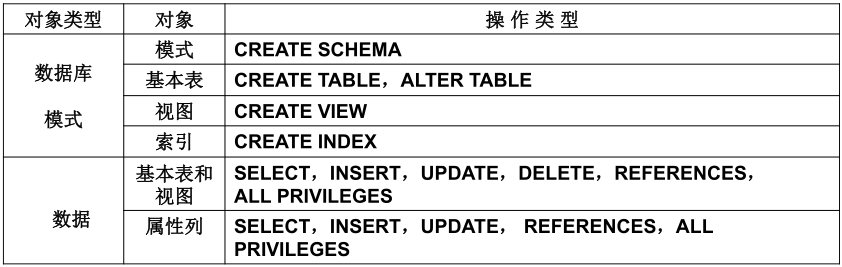
1. 自主存取控制方法

定义用户存取权限：定义**用户**可以**在哪些数据库对象**上进行**哪些类型的操作**

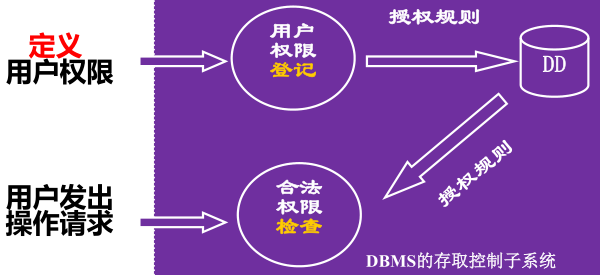
定义存取权限称为授权

用户权限的组成：数据对象、操作类型

通过 SQL 的GRANT 语句和REVOKE 语句实现



1. 授权：授予与回收



1. GRANT
   1. 语义：将对**指定操作对象**的**指定操作权限**授予**指定的用户**

GRANT语句的一般格式：

GRANT <权限>[,<权限>]...

ON <对象类型> <对象名>[,<对象类型> <对象名>]…

TO <用户>[,<用户>]...

[WITH GRANT OPTION];

对属性列的授权时必须明确指出相应属性列名

示例：把查询Student表和修改学生学号的权限授给用户U4

GRANT UPDATE(Sno), SELECT

ON Student

TO U4;

* 1. 发出GRANT

数据库管理员（超级用户DBA windows & sa）

数据库对象创建者（即属主Owner）

拥有该权限的用户（with grant option）

* 1. 按受权限的用户

一个或多个具体用户

PUBLIC（即全体用户）

* 1. WITH GRANT OPTION子句:

指定：可以再授予

没有指定：不能传播

**不允许**循环授权（a->b, b->c, c->a）

示例：把对表SC的INSERT权限授予U5用户，并允许他再将此权限授予其他用户

GRANT INSERT

ON SC

TO U5

WITH GRANT OPTION;

执行示例后，U5不仅拥有了对表SC的INSERT权限， 还可以传播此权限：

GRANT INSERT

ON SC

TO U6

WITH GRANT OPTION;

同样，U6还可以将此权限授予U7

1. REVOKE
   1. 授予的权限可以由**数据库管理员（不管是谁的都收回来）**或**其他授权者（只能收它自己的）**用REVOKE语句收回

REVOKE语句的一般格式为：

REVOKE <权限>[,<权限>]...

ON <对象类型> <对象名>[,<对象类型><对象名>]…

FROM <用户>[,<用户>]...[CASCADE | RESTRICT];

示例：把用户U4修改学生学号的权限收回

REVOKE UPDATE(Sno)

ON Student

FROM U4;

示例：收回所有用户对表SC的查询权限

REVOKE SELECT

ON SC

FROM PUBLIC;

* 1. **将用户U5的INSERT权限收回的时候应该使用CASCADE（凡是它U5分配的权限也被收回），否则拒绝执行该语句**

示例：把用户U5对SC表的INSERT权限收回

REVOKE INSERT

ON SC

FROM U5 CASCADE ;

如果U6或U7还从**其他用户**处获得对SC表的INSERT权限，则他们仍具有此权限，系统**只收回直接或间接从U5处获得的权限**

U1授予U2的SC表的select权限，U3也授予U2的SC表的select权限，此时收回U1的select权限（cascade），U2不具有U1授予U2的SC表的select权限，而具有U3授予U2的SC表的select权限

U4授予U5的SC表的select权限，超级用户（windows & sa）也授予U5的SC表的select权限，此时超级用户收回U5的select权限，U5就不具有SC表的select权限了

1. 小结
   1. 数据库管理员

**拥有所有对象的所有权限（可以收回所有用户的权限）**

根据实际情况不同的权限授予不同的用户

* 1. 用户

拥有**自己建立的对象**的全部的操作权限

可以使用GRANT，把权限授予其他用户

* 1. 被授权的用户

如果具有“继续授权”的许可，可以把获得的权限再授予其他用户

所有授予出去的权力在必要时又都可用REVOKE语句收回

1. 创建数据库模式的权限

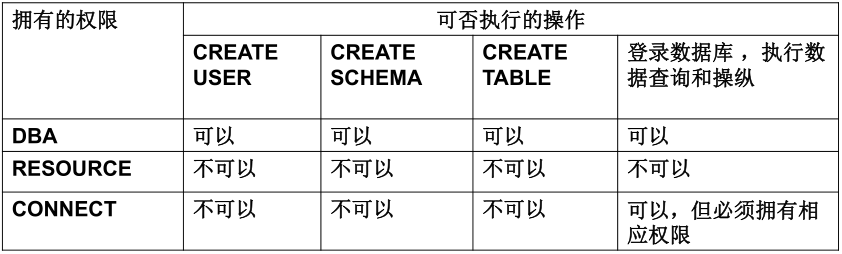
数据库管理员在创建用户时实现

CREATE USER语句格式

CREATE USER <username>

[WITH][DBA|RESOURCE|CONNECT];//默认是CONNECT

注： CREATE USER不是SQL标准，各个系统的实现相差甚远



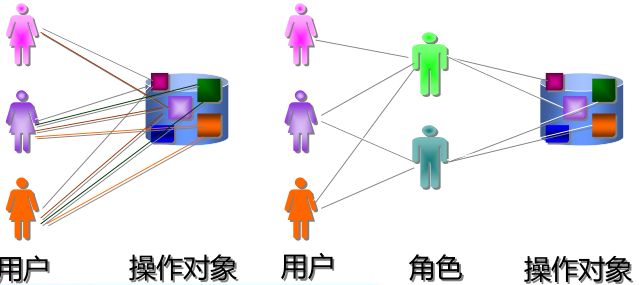
1. 数据库角色

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

简化授权的过程



1. 角色的创建

CREATE ROLE <角色名>

示例：首先创建一个角色 R1

CREATE ROLE R1;

1. 给角色授权

GRANT <权限>[,<权限>]…

ON <对象类型>对象名

TO <角色>[,<角色>]…

在角色当中SQLserver与标准语言不一样

Exec sp\_addrolemember r1, u1 角色给某个用户

Exec sp\_droprolemember r1, u1 收回

示例：然后使用GRANT语句，使角色R1拥有Student表的 SELECT、UPDATE、INSERT权限

GRANT SELECT, UPDATE, INSERT

ON Student

TO R1;

1. 将一个角色授予其他的角色或用户

GRANT <角色1>[,<角色2>]…

TO <角色3>[,<用户1>]…

[WITH ADMIN OPTION]

该语句把角色授予某**用户**，或授予另一个**角色**

授予者是角色的创建者或拥有在这个角色上的ADMIN OPTION

指定了WITH ADMIN OPTION则获得某种权限的**角色或用户**还可以把这种权限授予其他**角色**（角色1可以给角色2授予角色，角色1可以给用户1授予角色，用户1可以给角色1授予角色）

一个角色的权限：直接授予这个角色的全部权限加上其他角色授予这个角色的全部权限

示例：将这个角色授予U1,U2,U3 。使他们具有角色R1所包含的全部权限

GRANT R1

TO U1,U2,U3;

1. 角色权限的收回

REVOKE <权限>[,<权限>]…

ON <对象类型> <对象名>

FROM <角色>[,<角色>]…

**用户**可以**回收角色的权限**，从而修改角色拥有的权限

REVOKE执行者是**角色的创建者**和**拥有在这个（些）角色上的ADMIN OPTION**

示例：可以一次性通过R1来回收U1的这3个权限

REVOKE R1

FROM U1;

示例：角色的权限修改（使角色R1在原来的基础上增加了Student表的DELETE 权限）

GRANT DELETE

ON Student

TO R1;

1. 强制存取控制方法
2. 自主存取控制缺点

可能存在数据的“无意泄露”

原因：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

解决：对系统控制下的所有主客体实施强制存取控制策略

1. 强制存取控制方法
   1. 强制存取控制（MAC）

保证更高程度的安全性

用户不能直接感知或进行控制

适用于对数据有严格而固定密级分类的部门（军事、政府）

* 1. 主体与客体

在强制存取控制中，数据库管理系统所管理的全部实体被分为主体和客体两大类

**主体**是系统中的活动实体：数据库管理系统所管理的实际用户，代表用户的各进程

**客体**是系统中的被动实体，受主体操纵：文件、基本表、索引、视图

* 1. 敏感度标记

对于主体和客体，DBMS为它们每个实例（值）指派一个敏感度标记

敏感度标记分成若干级别

绝密（Top Secret，TS） >= 机密（Secret，S） >= 可信（Confidential，C） >= 公开（Public，P）

主体的敏感度标记称为许可证级别

客体的敏感度标记称为密级

* 1. 强制存取控制规则

仅当主体的许可证级别**大于或等于**客体的密级时，该主体才能读取相应的客体

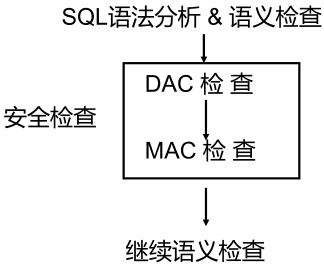
仅当主体的许可证级别**小于或等于**客体的密级时，该主体才能写相应的客体

* 1. 总结

强制存取控制（MAC）是对数据本身进行密级标记，无论数据如何复制，标记与数据是一个不可分的整体，只有符合密级标记要求的用户才可以操纵数据

实现强制存取控制时要首先实现自主存取控制，因为较高安全性级别提供的安全保护要包含较低级别的所有保护

自主存取控制与强制存取控制共同构成数据库管理系统的安全机制



先进行 自主存取控制 检查，通过 自主存取控制 检查的数据对象再由系统进行强制存取控制 检查，只有通过 强制存取控制 检查的数据对象方可存取

1. 视图机制

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

视图机制更主要的功能在于提供数据独立性，其安全保护功能太不精细，往往远不能达到应用系统的要求

间接实现了支持存取谓词的用户权限定

示例：建立计算机系学生的视图，把对该视图的SELECT限授于王平，把该视图上的所有操作权限授于张明

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’；

在视图上进一步定义存取权限

GRANT SELECT ON CS\_Student

TO 王平 ；

GRANTALL PRIVILIGES ON CS\_Student

TO 张明；

1. 审计

审计日志：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

·AUDIT语句：设置审计功能

示例：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；

·NOAUDIT语句：取消审计功能

示例：取消对SC表的一切审计

NOAUDIT ALTER，UPDATE ON SC；

1. 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（术语为明文，Plain text）变换为不可直接识别的格式（术语为密文，Cipher text）

加密方法：替换方法、置换方法、混合方法

1. 统计数据库安全性