数据库系统概论

An Introduction to Database System

第四章　数据库安全性

内蒙古农业大学计算机与信息工程学院

数据库安全性

- 问题的提出
  - 数据库的一大特点是数据可以共享
  - 但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（续）

- 数据库中数据的共享是在DBMS统一的严格的控制之下的共享，即只允许有合法使用权限的用户访问允许他存取的数据
- 数据库系统的安全保护措施是否有效是数据库系统主要的性能指标之一

4.1 数据库安全性概论

- 什么是数据库的安全性
  - 数据库的安全性是指保护数据库，防止因用户非法使用数据库造成数据泄露、更改或破坏。
- 什么是数据的保密
  - 数据保密是指用户合法地访问到机密数据后能否对这些数据保密。
  - 通过制订法律道德准则和政策法规来保证。

4.1 数据库安全性概论

- 什么是计算机系统的安全性
  - 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，数据遭到更改或泄露等。

4.1.1 数据库的不安全因素

- **非授权用户对数据库的恶意存取和破坏**
  - 一些黑客（**Hacker**）和犯罪分子在用户存取数据库时猎取用户名和用户口令，然后假冒合法用户偷取、修改甚至破坏用户数据。
  - 数据库管理系统提供的安全措施主要包括用户身份鉴别、存取控制和视图等技术。

4.1.1 数据库的不安全因素

- **数据库中重要或敏感的数据被泄露**
  - 黑客和敌对分子千方百计盗窃数据库中的重要数据，一些机密信息被暴露。
  - 数据库管理系统提供的主要技术有强制存取控制、数据加密存储和加密传输等。
  - 审计日志分析

4.1.1 数据库的不安全因素

- 安全环境的脆弱性
  - 数据库的安全性与计算机系统的安全性紧密联系
    - 计算机硬件、操作系统、网络系统等的安全性
  - 建立一套可信计算机系统的概念和标准

4.1.2 安全标准简介

- 1985年美国国防部（DoD）正式颁布《 DoD可信计算机系统评估标准》（简称TCSEC或DoD85）
  - TCSEC又称桔皮书
  - TCSEC标准的目的
    - 提供一种标准，使用户可以对其计算机系统内敏感信息安全操作的可信程度做评估。
    - 给计算机行业的制造商提供一种可循的指导规则，使其产品能够更好地满足敏感应用的安全需求。

4.1.2 安全标准简介

- 不同国家建立在TCSEC概念上的评估准则
  - 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）

4.1.2 安全标准简介

- **1993年，CTCPEC、FC、TCSEC和ITSEC**联合行动，解决原标准中概念和技术上的差异，称为**CC（Common Criteria）**项目
- **1999年 CC V2.1**版被**ISO**采用为国际标准
  2001年 CC V2.1版被我国采用为国家标准
- 目前**CC**已基本取代了**TCSEC**，成为评估信息产品安全性的主要标准。

TCSEC标准

- **1991年4月美国NCSC（国家计算机安全中心）颁布了《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 简称TDI）**
- **TDI又称紫皮书。它将TCSEC扩展到数据库管理系统**
- **TDI中定义了数据库管理系统的设计与实现中需满足和用以进行安全性级别评估的标准**

TCSEC标准

- TDI/TCSEC标准的基本内容
  - TDI与TCSEC一样，从*四个方面*来描述安全性级别划分的指标
    - 安全策略
    - 责任
    - 保证
    - 文档

TCSEC标准

- R1 安全策略（Security Policy）

  R1.1 自主存取控制 （Discretionary Access Control，简记为DAC）

  R1.2 客体重用（Object Reuse）

  R1.3 标记（Labels）

  R1.4 强制存取控制（Mandatory Access Control，简记为MAC）

TCSEC标准

- R2 责任（Accountability）

  R2.1 标识与鉴别（Identification & Authentication）

  R2.2 审计（Audit）

- R3 保证（Assurance）

  R3.1 操作保证（Operational Assurance）

  R3.2 生命周期保证（Life Cycle Assurance）

TCSEC标准

- R4 文档（Documentation）

  R4.1 安全特性用户指南（Security Features User's Guide）

  R4.2 可信设施手册（Trusted Facility Manual）

  R4.3 测试文档（Test Documentation）

  R4.4 设计文档（Design Documentation）

TCSEC/TDI安全级别划分

- TCSEC/TDI安全级别划分

安全级别	定　　义
A1	验证设计（Verified Design）
B3	安全域（Security Domains）
B2	结构化保护（Structural Protection）
B1	标记安全保护（Labeled Security Protection）
C2	受控的存取保护（Controlled Access Protection）
C1	自主安全保护（Discretionary Security Protection）
D	最小保护（Minimal Protection）

TCSEC/TDI安全级别划分

- 四组(division)七个等级
  - D
  - C（C1，C2）
  - B（B1，B2，B3）
  - A（A1）
- 按系统可靠或可信程度逐渐增高
- 各安全级别之间具有一种偏序向下兼容的关系，即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要求，同时提供更多或更完善的保护能力。

TCSEC/TDI安全级别划分

- D级
  - 将一切不符合更高标准的系统均归于D组
  - 典型例子：DOS是安全标准为D的操作系统
    - DOS在安全性方面几乎没有什么专门的机制来保障

TCSEC/TDI安全级别划分

- C1级
  - 非常初级的自主安全保护
  - 能够实现对用户和数据的分离，进行自主存取控制（DAC），保护或限制用户权限的传播。

TCSEC/TDI安全级别划分

- C2级
  - 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
  - 达到C2级的产品在其名称中往往不突出“安全”(Security)这一特色

TCSEC/TDI安全级别划分

- 典型例子
  - 操作系统
    - Microsoft的Windows NT 3.5，
    - 数字设备公司的Open VMS VAX 6.0和6.1
  - 数据库
    - Oracle公司的Oracle 7
    - Sybase公司的 SQL Server 11.0.6

TCSEC/TDI安全级别划分

- B1级
  - 标记安全保护。“安全”(Security)或“可信的”(Trusted)产品。
  - 对系统的数据加以标记，对标记的主体和客体实施强制存取控制（MAC）、审计等安全机制

TCSEC/TDI安全级别划分

- 典型例子
  - 操作系统
    - 数字设备公司的SEVMS VAX Version 6.0
    - 惠普公司的HP-UX BLS release 9.0.9+
  - 数据库
    - Oracle公司的Trusted Oracle 7
    - Sybase公司的Secure SQL Server version 11.0.6
    - Informix公司的Incorporated INFORMIX-OnLine / Secure 5.0

TCSEC/TDI安全级别划分

- B2级
  - 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。
  - 经过认证的B2级以上的安全系统非常稀少

TCSEC/TDI安全级别划分

- 典型例子
  - 操作系统
    - 只有Trusted Information Systems公司的Trusted XENIX一种产品
  - 标准的网络产品
    - 只有Cryptek Secure Communications公司的LLC VSLAN一种产品
  - 数据库
    - 没有符合B2标准的产品

TCSEC/TDI安全级别划分

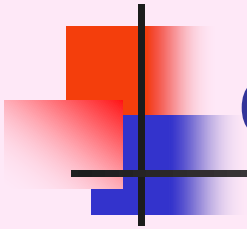
- B3级
  - 安全域。
  - 该级的TCB必须满足访问监控器的要求，审计跟踪能力更强，并提供系统恢复过程。

TCSEC/TDI安全级别划分

- A1级
  - 验证设计，即提供B3级保护的同时给出系统的形式化设计说明和验证以确信各安全保护真正实现。

CC

- CC
- 提出国际公认的表述信息技术安全性的结构
- 把信息产品的安全要求分为
  - 安全功能要求
  - 安全保证要求



CC

- CC文本组成
- 简介和一般模型
  - 有关术语、基本概念和一般模型以及与评估有关的一些框架
- 安全功能要求
  - 列出了一系列类、子类和组件
- 安全保证要求
  - 列出了一系列保证类、子类和组件
  - 提出了评估保证级（从EAL1至EAL7共分为七级）

CC

- CC评估保证级（EAL）划分

评估保证级	定义	TCSEC安全级别（近似相当）
EAL1	功能测试（**functionallytested**）	
EAL2	结构测试（**structurallytested**）	**C1**
EAL3	系统地测试和检查（**methodicallytestedand checked**）	**C2**
EAL4	系统地设计、测试和复查（**methodicallydesigned，tested，andreviewed**）	**B1**
EAL5	半形式化设计和测试（**semiformallydesignedand tested**）	**B2**
EAL6	半形式化验证的设计和测试（**semiformallyverified designandtested**）	**B3**
EAL7	形式化验证的设计和测试（**formallyverified designandtested**）	**A1**

4.2 数据库安全性控制

- 非法使用数据库的情况
  - 用户编写一段合法的程序绕过DBMS及其授权机制，通过操作系统直接存取、修改或备份数据库中的数据
  - 直接或编写应用程序执行非授权操作

数据库安全性控制（续）

- 通过多次合法查询数据库从中推导出一些保密数据

  例：某数据库应用系统禁止查询单个人的工资，但允许查任意一组人的平均工资。用户甲想了解张三的工资，于是他：

  首先查询包括张三在内的一组人的平均工资

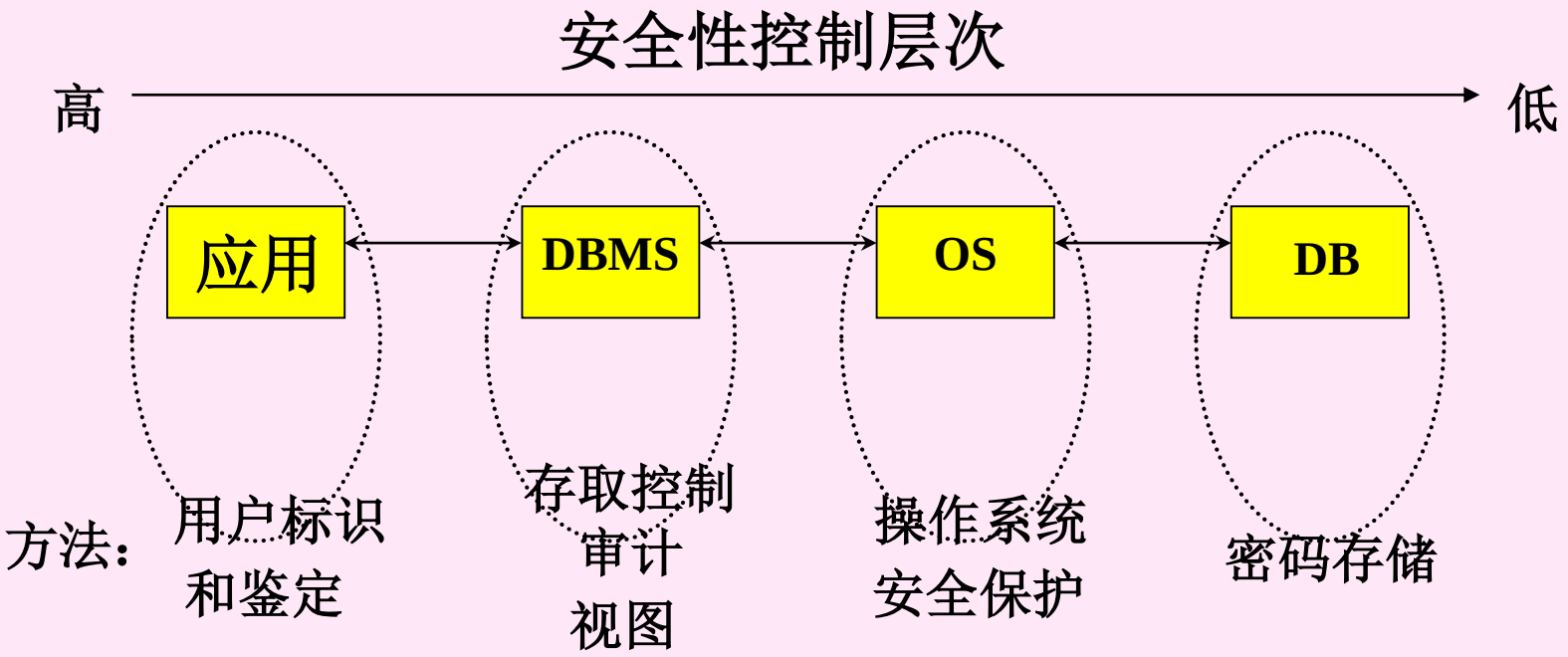
  然后查用自己替换张三后这组人的平均工资

  从而推导出张三的工资

- 破坏安全性的行为可能是无意的，故意的，恶意的。

数据库安全性控制（续）

◆计算机系统中，安全措施是一级一级层层设置

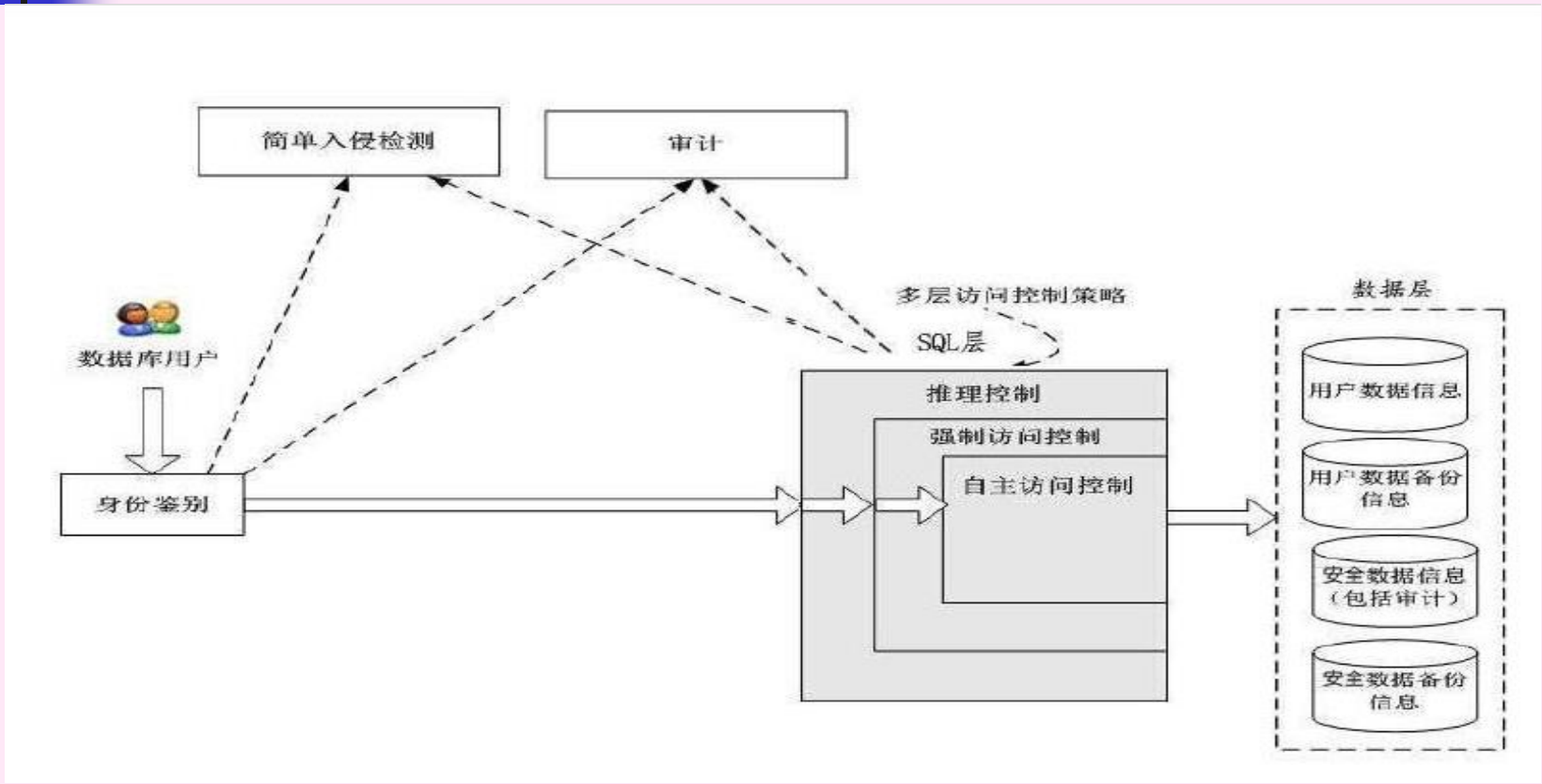


计算机系统的安全模型

数据库安全性控制（续）

- 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

数据库安全性控制（续）



数据库管理系统安全性控制模型

数据库安全性控制（续）

- 数据库安全性控制的常用方法
  - 用户身份鉴别
  - 存取控制
  - 视图
  - 审计
  - 密码存储

4.2.1 用户身份鉴别

- 用户身份鉴别
  - 数据库管理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）

4.2.1 用户身份鉴别

基本方法

- 系统提供一定的方式让用户标识自己的名字或身份；
- 系统内部记录着所有合法用户的标识；
- 每次用户要求进入系统时，由系统核对用户提供的身份标识；
- 通过鉴定后才提供机器使用权。
- 用户标识和鉴定可以重复多次

用户身份鉴别的常用方法

- 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
- 生物特征鉴别
  - 通过生物特征进行认证的技术，如指纹、虹膜和掌纹等
- 智能卡鉴别
  - 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

4.2.2 存取控制

- 存取控制机制的组成
- 定义用户存取权限
  - 用户对某一数据对象的操作权力称为权限
  - DBMS提供适当的语言来定义用户权限，这些定义被编译后被存储在数据字典中，称作安全规则或授权规则
- 用户合法权限检查
  - 用户发出存取数据库操作请求
  - DBMS查找数据字典，根据安全规则进行合法权限检查
  - 对于通过鉴定获得权限的用户（即合法用户），系统根据他的存取权限定义对他的各种操作请求进行控制，确保他只执行合法操作。
- 用户权限定义和合法权限检查机制一起组成了DBMS的**存取控制子系统**

存取控制（续）

- 常用存取控制方法
  - 自主存取控制（Discretionary Access Control，简称DAC）
    - C2级
    - 同一用户对于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也有不同的权限
    - 用户还可将其拥有的存取权限转授给其他用户
    - 灵活

存取控制（续）

- 常用存取控制方法
  - 强制存取控制（Mandatory Access Control，简称MAC）
    - B1级
    - 严格
    - 每一个数据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

4.2.3 自主存取控制方法

- 通过SQL的**GRANT**语句和**REVOKE**语句实现
- 用户权限
  - 数据对象
  - 操作类型
- 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**

自主存取控制方法（续）

- 关系数据库系统中的存取控制对象

对象类型	对象	操作类型
数据库模式	模式	**CREATESCHEMA**
	基本表	**CREATETABLE，ALTERTABLE**
	视图	**CREATE VIEW**
	索引	**CREATE INDEX**
数据	基本表和视图	**SELECT，INSERT，UPDATE，DELETE，REFERENCES，ALLPRIVILEGES**
	属性列	**SELECT，INSERT，UPDATE，REFERENCES，ALLPRIVILEGES**

4.2.4 授权：授予与回收

1.GRANT

❖ **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];

❖ 语义：将对指定操作对象的指定操作权限授予指定的用户

4.2.4 授权：授予与回收

- 发出**GRANT**:
  - 数据库管理员
  - 数据库对象创建者（即属主**Owner**）
  - 拥有该权限的用户
- 按受权限的用户
  - 一个或多个具体用户
  - **PUBLIC**（即全体用户）

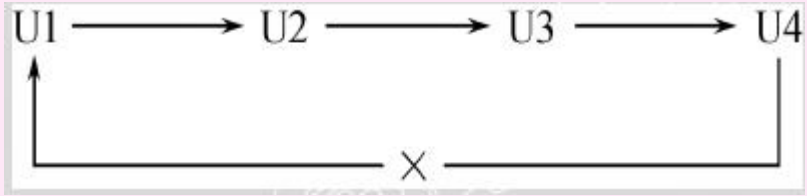
4.2.4 授权：授予与回收

❖**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



4.2.4 授权：授予与回收

[例4.1] 把查询**Student**表权限授给用户**U1**

```
GRANT   SELECT
ON   TABLE   Student
TO   U1;
```

4.2.4 授权：授予与回收

[例4.2] 把对**Student**表和**Course**表的全部权限授予用户**U2**和**U3**

```
GRANT ALL PRIVILIGES
ON TABLE Student,Course
TO U2,U3;
```

4.2.4 授权：授予与回收

[例4.3] 把对表**SC**的查询权限授予所有用户

```
GRANT SELECT
ON TABLE SC
TO PUBLIC;
```

4.2.4 授权：授予与回收

[例4.4] 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 TABLE Student
TO U4;
```

❖对属性列的授权时必须明确指出相应属性列名

4.2.4 授权：授予与回收

[例4.5] 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 TABLE SC
TO U5
WITH GRANT OPTION;
```

传播权限

执行例**4.5**后，**U5**不仅拥有了对表**SC**的**INSERT**权限，还可以传播此权限：

[例4.6] GRANT INSERT
ON TABLE SC
TO U6
WITH GRANT OPTION;

同样，**U6**还可以将此权限授予**U7**：

[例4.7] GRANT INSERT
ON TABLE SC
TO U7;

但**U7**不能再传播此权限。

传播权限

执行了例**4.1~**例**4.7**语句后学生-课程数据库中的用户权限定义表

授权用户名	被授权用户名	数据库对象名	允许的操作类型	能否转授权
DBA	**U1**	关系**Student**	**SELECT**	不能
DBA	**U2**	关系**Student**	**ALL**	不能
DBA	**U2**	关系**Course**	**ALL**	不能
DBA	**U3**	关系**Student**	**ALL**	不能
DBA	**U3**	关系**Course**	**ALL**	不能
DBA	**PUBLIC**	关系**SC**	**SELECT**	不能
DBA	**U4**	关系**Student**	**SELECT**	不能
DBA	**U4**	属性列 **Student.Sno**	**UPDATE**	不能
DBA	**U5**	关系**SC**	**INSERT**	能
U5	**U6**	关系**SC**	**INSERT**	能
U6	**U7**	关系**SC**	**INSERT**	不能

4.2.4 授权：授予与回收

2.REVOKE

❖授予的权限可以由数据库管理员或其他授权者用**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];

4.2.4 授权：授予与回收

[例4.8] 把用户**U4**修改学生学号的权限收回

```
REVOKE UPDATE(Sno)
ON TABLE Student
FROM U4;
```

4.2.4 授权：授予与回收

[例4.9] 收回所有用户对表**SC**的查询权限

```
REVOKE SELECT
ON TABLE SC
FROM PUBLIC;
```

4.2.4 授权：授予与回收

[例4.10] 把用户**U5**对**SC**表的**INSERT**权限收回

```
REVOKE INSERT
ON TABLE SC
FROM U5 CASCADE;
```

- 将用户**U5**的**INSERT**权限收回的时候应该使用CASCADE，否则拒绝执行该语句
- 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

4.2.4 授权：授予与回收

执行例**4.8~4.10**语句后学生-课程数据库中的用户权限定义表

授权用户名	被授权用户名	数据库对象名	允许的操作类型	能否转授权
DBA	**U1**	关系**Student**	**SELECT**	不能
DBA	**U2**	关系**Student**	**ALL**	不能
DBA	**U2**	关系**Course**	**ALL**	不能
DBA	**U3**	关系**Student**	**ALL**	不能
DBA	**U3**	关系**Course**	**ALL**	不能
DBA	**U4**	关系**Student**	**SELECT**	不能

4.2.4 授权：授予与回收

- 数据库管理员：
  - 拥有所有对象的所有权限
  - 根据实际情况不同的权限授予不同的用户
- 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
- 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

4.2.4 授权：授予与回收

◆ 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];

注：

CREATE USER不是**SQL**标准，各个系统的实现相差甚远

4.2.4 授权：授予与回收

- **CREATE USER**语句格式说明
  - 只有系统的超级用户才有权创建一个新的数据库用户
  - 新创建的数据库用户有三种权限：**CONNECT**、**RESOURCE**和**DBA**
  - 如没有指定创建的新用户的权限，默认该用户拥有**CONNECT**权限。拥有**CONNECT**权限的用户不能创建新用户，不能创建模式，也不能创建基本表，只能登录数据库

4.2.4 授权：授予与回收

- **CREATE USER**语句格式说明（续）
  - 拥有**RESOURCE**权限的用户能创建基本表和视图，成为所创建对象的属主。但不能创建模式，不能创建新的用户
  - 拥有**DBA**权限的用户是系统中的超级用户，可以创建新的用户、创建模式、创建基本表和视图等；**DBA**拥有对所有数据库对象的存取权限，还可以把这些权限授予一般用户

4.2.4 授权：授予与回收

拥有的权限	可否执行的操作			
	CREATE USER	**CREATE SCHEMA**	**CREATE TABLE**	登录数据库，执行数据查询和操纵
DBA	可以	可以	可以	可以
RESOURCE	不可以	不可以	不可以	不可以
CONNECT	不可以	不可以	不可以	可以，但必须拥有相应权限

权限与可执行的操作对照表

4.2.5 数据库角色

- 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

4.2.5 数据库角色

- 角色的创建

  CREATE ROLE <角色名>

- 给角色授权

  GRANT <权限>[,<权限>]…

  ON <对象类型>对象名

  TO <角色>[,<角色>]…

4.2.5 数据库角色

◆ 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

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

4.2.5 数据库角色

◆ 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
- ●拥有在这个（些）角色上的**ADMIN OPTION**

4.2.5 数据库角色

[例4.11] 通过角色来实现将一组权限授予一个用户。

步骤如下：

（1）首先创建一个角色 **R1**

```
CREATE  ROLE  R1;
```

（2）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 TABLE Student
TO R1;
```

4.2.5 数据库角色

（3）将这个角色授予王平，张明，赵玲。使他们具有角色**R1**所包含的全部权限

GRANT R1

TO 王平,张明,赵玲;

（4） 可以一次性通过**R1**来回收王平的这**3**个权限

REVOKE R1

FROM 王平;

4.2.5 数据库角色

[例4.12] 角色的权限修改

```
GRANT DELETE
ON TABLE Student
TO R1;
```

使角色**R1**在原来的基础上增加了**Student**表的**DELETE** 权限

4.2.5 数据库角色

[例4.13]

REVOKE SELECT

ON TABLE Student

FROM R1;

使**R1**减少了**SELECT**权限

4.2.6 强制存取控制方法

- 自主存取控制的缺点
  - 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

4.2.6 强制存取控制方法

- 什么是强制存取控制
  - 强制存取控制(MAC)是指系统为保证更高程度的安全性，按照TDI/TCSEC标准中安全策略的要求，所采取的强制存取检查手段。
  - MAC不是用户能直接感知或进行控制的。
  - MAC适用于对数据有严格而固定密级分类的部门
    - 军事部门
    - 政府部门

强制存取控制方法（续）

- 主体与客体
  - 在MAC中，DBMS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
    - DBMS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，是受主体操纵的
    - 文件
    - 基表
    - 索引
    - 视图

强制存取控制方法（续）

- 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（Label）
  - 敏感度标记分成若干级别
    - 绝密（Top Secret）
    - 机密（Secret）
    - 可信（Confidential）
    - 公开（Public）
    - TS>=S>=C>=P

强制存取控制方法（续）

- 主体的敏感度标记称为许可证级别（Clearance Level）
- 客体的敏感度标记称为密级（Classification Level）
- MAC机制就是通过对比主体的Label和客体的Label，最终确定主体是否能够存取客体

强制存取控制方法（续）

- 强制存取控制规则
  - 当某一用户（或某一主体）以标记label注册入系统时，系统要求他对任何客体的存取必须遵循下面两条规则：

（1）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；

（2）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。

强制存取控制方法（续）

- 修正规则：
  - 主体的许可证级别 <=客体的密级
    主体能写客体
  - 用户可为写入的数据对象赋予高于自己的许可证级别的密级
  - 一旦数据被写入，该用户自己也不能再读该数据对象了。

强制存取控制方法（续）

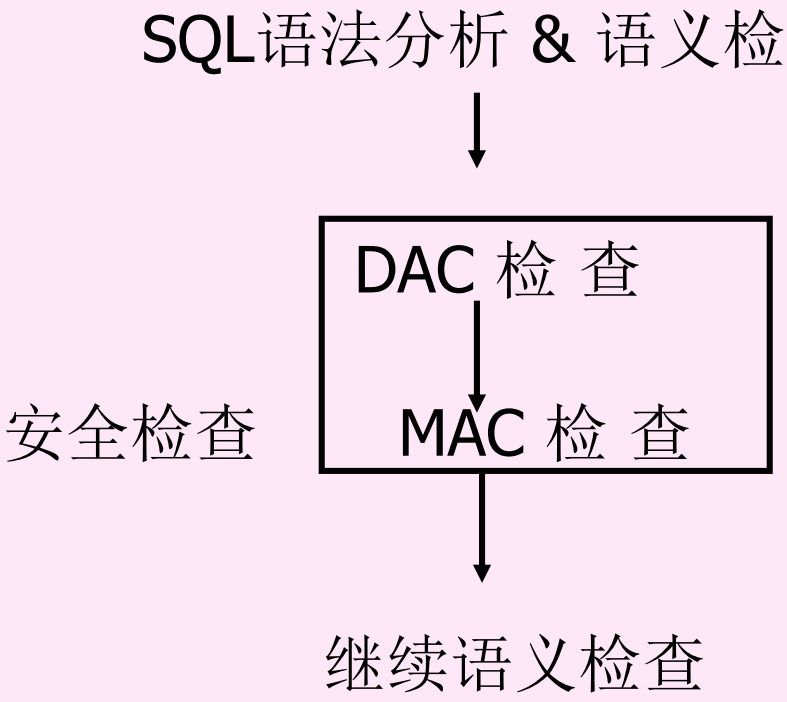
- 强制存取控制的特点
  - MAC是对数据本身进行密级标记
  - 无论数据如何复制，标记与数据是一个不可分的整体
  - 只有符合密级标记要求的用户才可以操纵数据
  - 从而提供了更高级别的安全性

MAC与DAC

- 实现强制存取控制时要首先实现自主存取控制
  - 原因：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DAC与强制存取控制MAC共同构成DBMS的安全机制

强制存取控制方法（续）

DAC + MAC安全检查示意图



先进行DAC检查，通过DAC检查的数据对象再由系统进行MAC检查，只有通过MAC检查的数据对象方可存取。

4.3 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。

视图机制（续）

- 视图机制与授权机制配合使用:
- 首先用视图机制屏蔽掉一部分保密数据
- 视图上面再进一步定义存取权限
- 间接实现了支持存取谓词的用户权限定义

视图机制（续）

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';
```

视图机制（续）

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;

4.4 审计

- 什么是审计
  - 启用一个专用的审计日志（Audit Log）
  - 将用户对数据库的所有操作自动记录下来放入审计日志
  - 审计员可以利用审计日志监控数据库中的各种行为，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
  - C2以上安全级别的DBMS必须具有审计功能

审计（续）

- 审计功能的可选性
  - 审计很费时间和空间
  - DBA可以根据应用对安全性的要求，灵活地打开或关闭审计功能
  - 审计功能主要用于安全性要求较高的部门

审计（续）

- 强制性机制:

  用户识别和鉴定、存取控制、视图

- 预防监测手段:

  审计技术

4.5 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），从而使得不知道解密算法的人无法获知数据的内容。

数据加密（续）

- 加密方法
  - 存储加密
    - 透明存储加密：
      - 内核级加密保护方式，对用户完全透明
      - 将数据在写到磁盘时对数据进行加密，授权用户读取数据时再对其进行解密
      - 数据库的应用程序不需要做任何修改，只需在创建表语句中说明需加密的字段即可

      基于数据库内核的数据存储加密、解密方法：性能较好，安全完备性较高
    - 非透明存储加密：
      - 通过多个加密函数实现

数据加密（续）

- 加密方法
  - 传输加密：防止数据被网络恶意用户截获或篡改
    - 链路加密
      - 在链路层进行加密
      - 传输信息由报头和报文两部分组成
      - 报文和报头均加密
    - 端到端加密
      - 在发送端加密，接收端解密
      - 只加密报文不加密报头
      - 所需密码设备数量相对较少，容易被非法监听者发现并从中获取敏感信息

4.6 其他安全性保护

- 推理控制
  - 处理强制存取控制未解决的问题
  - 避免用户利用能够访问的数据推知更高密级的数据
  - 常用方法
    - 基于函数依赖的推理控制
    - 基于敏感关联的推理控制
- 隐蔽信道
  - 处理强制存取控制未解决的问题

4.6 其他安全性保护

- 数据隐私保护
  - 描述个人控制其不愿他人知道或他人不便知道的个人数据的能力
  - 范围很广：数据收集、数据存储、数据处理和数据发布等各个阶段

4.7 SQL Server的安全性机制

就目前而言，绝大多数数据库管理系统都还是运行在某一特定操作系统平台下的应用程序，SQL Server也不例外。SQL Server的安全性机制可以划分为4个等级：

- 1、客户机操作系统的安全性；
- 2、SQL Server的登录安全性；
- 3、数据库的使用安全性；
- 4、数据库对象的使用安全性。

SQL Server登录模式

- SQL Server标准登录模式
- SQL Server集成登录模式
- 使用企业管理器建立登录账户

用户

- 在实现数据库的安全登录以后，检验用户权限的下一个安全等级就是数据库的访问权。数据库的访问权是通过映射数据库的用户与登录账户之间的关系来实现。数据库的用户是数据库级的安全实体，就像登录账户是服务器级的安全实体一样。
- 将登录账户添加为数据库用户后，使用该登录账户登录的SQL Server用户就可以实现对数据库的访问了
- SQL Server的数据库级别上也存在着两个特殊的数据库用户。分别是dbo和guest。 。

用户

- dbo是数据库的拥有者，在安装SQL Server时，被设置到model数据库中，而且不能被删除，所有dbo在每个数据库中都存在。dbo是数据库的最高权力拥有者，可以在数据库范围内执行一切操作。dbo永远无法从数据库中删除并且它的用户ID总是为1。
- dbo用户对应于创建该数据库的登录账户，所有系统数据库的dbo都对应于sa账户。
- guest用户可以使任何已经登录到SQL Server服务器的用户都可以访问数据库，所有的系统数据库除model以外都有guest用户。所有新建的数据库都没有这个用户，如果有必要添加guest用户，则必须使用系统存储过程明确的建立这个用户。

角色

- 角色是SQL Server 2000引进的用来集中管理数据库或服务器权限的概念。在SQL Server 2000中可以把某些用户设置成某一角色，当对角色进行权限设置时，这些用户自动继承该角色的权限。这样，只要对角色进行管理就可以实现对属于该角色的所有用户的权限管理，大大减少了工作量。角色的意义就好像现实生活中的官位。数据库管理员将操作数据库的权限赋予角色，就好像把职权赋予了一个官位。然后，数据库管理员可以将角色再赋给数据库用户或登录账户，从而使数据库用户或登录账户拥有了相应的权限。
- SQL Server中有两类角色：固定服务器角色和固定数据库角色。

角色

固定服务器角色

角色	描述
sysadmin	执行SQL Server中的任何操作
dbcreator	创建和修改数据库
diskadmin	管理磁盘文件
processadmin	管理系统进程
securityadmin	管理和审计登录账号
severadmin	配置服务器范围内的设置
setupadmin	可以增加、删除连接服务器；建立数据库复制；管理扩展的存储过程
bulkadmin	执行块插入操作

角色

固定数据库角色

角色	描述
public	维护全部默认的许可
db_owner	执行数据库中的任何操作，可以将对象权限指定给其他用户，其包括了以下各角色的所有权限
db_accessadmin	可以增加或删除Windows认证模式下的数据库用户或用户登录者
db_addladmin	增加、修改或删除数据库对象
db_securityadmin	执行语句和对象权限管理
db_datareader	检索任意表中的数据
db_datawriter	增加、修改和删除所有表中的数据
db_denydatareader	不能检索任意一个表中数据
db_denydatawriter	不能增加、修改或删除任意一个表中的数据

权限

- 在数据库内部，SQL Server提供权限（Permission）作为访问权设置的最后一道关卡。在SQL Server数据库里的每一个数据库对象都由一个该数据库的用户所拥有，拥有者以数据库赋予的userID作为标识。
- 当数据库对象刚刚创建完后，只有拥有者可以访问该数据库对象。任何其他用户想访问该对象必须首先获得拥有者赋予他们的权限。拥有者可以授予权限给指定的数据库用户。这种权限被称为对象权限。
- 通常权限可以分为三种类型：对象权限、命令权限和暗示性权限。

下课了。。。

认真

休息一会儿。。。