数据库系统概论

An Introduction to Database System

第四章 数据库安全性

福州大学软件学院

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
  - 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**

数据库安全性

- 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

数据库安全性

- 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。

第四章 数据库安全性

4.1 计算机安全性概述

4.1.1 计算机系统的三类安全性问题

4.1.2 安全标准简介

4.1.1 计算机系统的三类安全性问题

- 计算机系统安全性
  - 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。

计算机系统的三类安全性问题（续）

- 计算机安全涉及问题
  - 计算机系统本身的技术问题
    - 计算机安全理论与策略
    - 计算机安全技术
  - 管理问题
    - 安全管理
    - 安全评价
    - 安全产品

计算机系统的三类安全性问题（续）

- 计算机安全涉及问题(续)
  - 法学
    - 计算机安全法律
  - 犯罪学
    - 计算机犯罪与侦察
    - 安全监察
  - 心理学

计算机系统的三类安全性问题（续）

- 三类计算机系统安全性问题
  - 技术安全类
  - 管理安全类
  - 政策法律类

计算机系统的三类安全性问题（续）

- 技术安全
  - 指计算机系统中采用具有一定安全性的硬件、软件来实现对计算机系统及其所存数据的安全保护，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增加、不丢失、不泄露。

计算机系统的三类安全性问题（续）

- 管理安全
  - 软硬件意外故障、场地的意外事故、管理不善导致的计算机设备和数据介质的物理破坏、丢失等安全问题

计算机系统的三类安全性问题（续）

- 政策法律类
  - 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

4.1 计算机安全性概论

4.1.1 计算机系统的三类安全性问题

4.1.2 安全标准简介

4.1.2 安全标准简介

- 为降低进而消除对系统的安全攻击，各国引用或制定了一系列安全标准
  - TCSEC (桔皮书)
  - TDI (紫皮书)

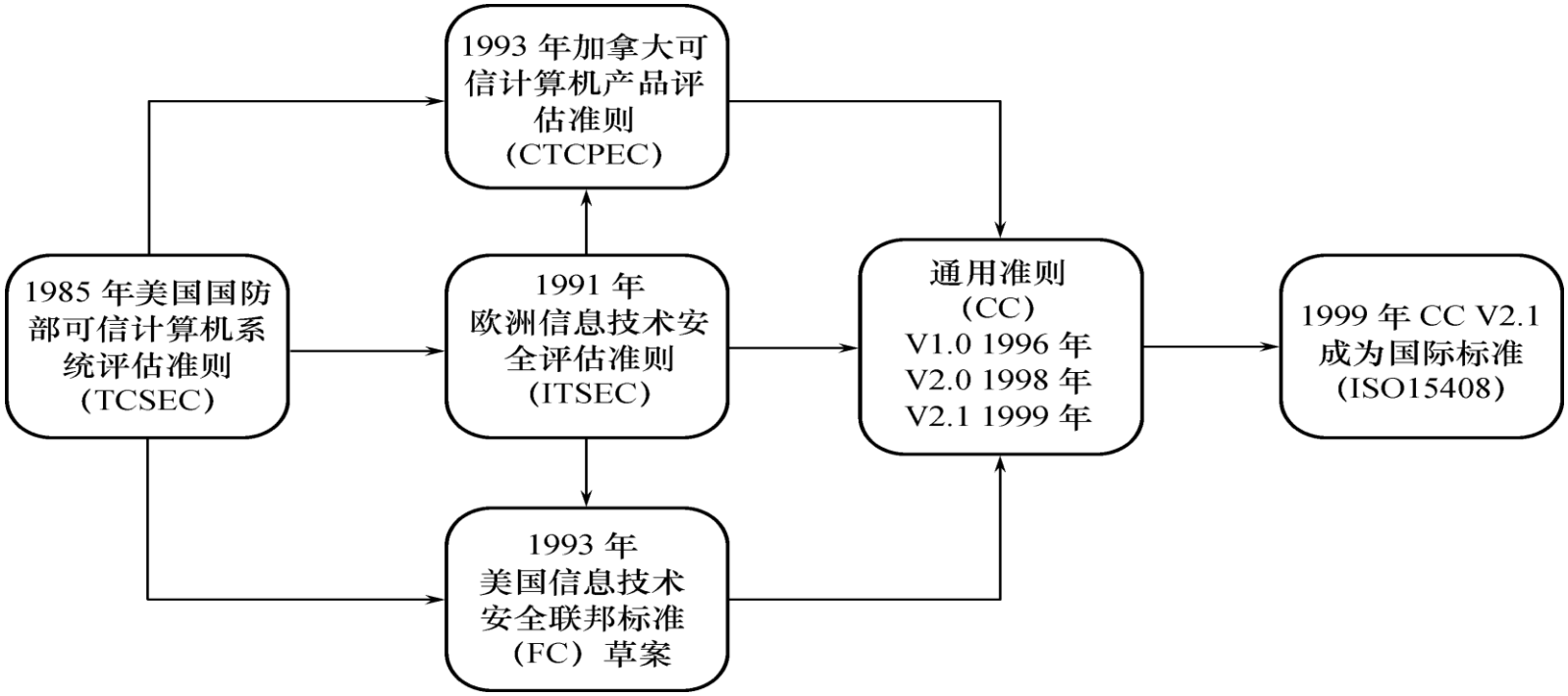
4.1.2 安全标准简介

- 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

- 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。

安全标准简介（续）



信息安全标准的发展历史

安全标准简介（续）

- TCSEC/TDI标准的基本内容
  - TCSEC/TDI，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/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（续）

- 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（续）

- 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（续）

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（续）

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（续）

- 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（续）

- 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（续）

- 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（续）

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（续）

- 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（续）

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（续）

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。

TCSEC/TDI安全级别划分（续）

- B2以上的系统
  - 还处于理论研究阶段
  - 应用多限于一些特殊的部门，如军队等
  - 美国正在大力发展安全产品，试图将目前仅限于少数领域应用的B2安全级别下放到商业应用中来，并逐步成为新的商业标准

TCSEC/TDI安全级别划分（续）

表 9.2　不同安全级别对安全指标的支持情况

	自主存取控制	客体重用	标记完整性	标记信息的扩散	主体敏感度标记	设备标记	强制存取控制	标识与鉴别	可信路径	审计	系统体系结构	系统完整性	屏蔽信道分析	可信设施管理	可信恢复	安全测试	设计规范和验证	配置管理	可信分配	安全特性用户指南	可信设施手册	测试文档	设计文档
C1	■							■			■	■				■				■	■	■	■
C2	▨	■						▨		■	■	▒				▨				▒	▨	▒	▒
B1	▒	▒	■	■			■	▨		▨	▨	▒				▨	■			▒	▨	▒	▨
B2	▒	▒	▒	▒	■	■	▨	▒	■	▨	▨	▒	■	■		▨	▨	■		▒	▨	▨	▨
B3	▨	▒	▒	▒	▒	▒	▒	▒	▨	▨	▨	▒	▨	▨	■	▨	▨	▒		▒	▨	▒	▨
A1	▒	▒	▒	▒	▒	▒	▒	▒	▒	▒	▒	▒	▨	▒	▒	▨	▨	▨	■	▒	▒	▨	▨

TCSEC/TDI安全级别划分（续）

- 表示该级不提供对该指标的支持；
- 表示该级新增的对该指标的支持；
- 表示该级对该指标的支持与相邻低一级的等级一样；
- 表示该级对该指标的支持较下一级有所增加或改动。

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（续）

- CC文本组成
  - 简介和一般模型
  - 安全功能要求
  - 安全保证要求

CC（续）

CC评估保证级划分

评估保证级	定　　义	TCSEC安全级别（近似相当）
EAL1	功能测试（functionally tested）	
EAL2	结构测试（structurally tested）	C1
EAL3	系统地测试和检查（methodically tested and checked）	C2
EAL4	系统地设计、测试和复查（methodically designed， tested， and reviewed）	B1
EAL5	半形式化设计和测试（semiformally designed and tested）	B2
EAL6	半形式化验证的设计和测试（semiformally verified design and tested)	B3
EAL7	形式化验证的设计和测试（formally verified design and tested）	A1

第四章 数据库安全性

4.1 计算机安全性概述

4.2 数据库安全性控制

4.3 视图机制

4.4 审计（Audit）

4.5 数据加密

4.6 统计数据库安全性

4.7 小结

4.2 数据库安全性控制概述

- 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；

4.2 数据库安全性控制概述

- 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

Z+N=X

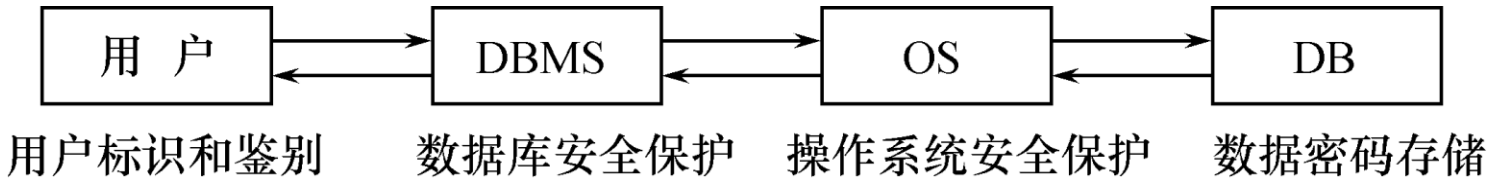
B+N=Y

B=Y-(X-Z)

- 破坏安全性的行为可能是无意的，故意的，恶意的。

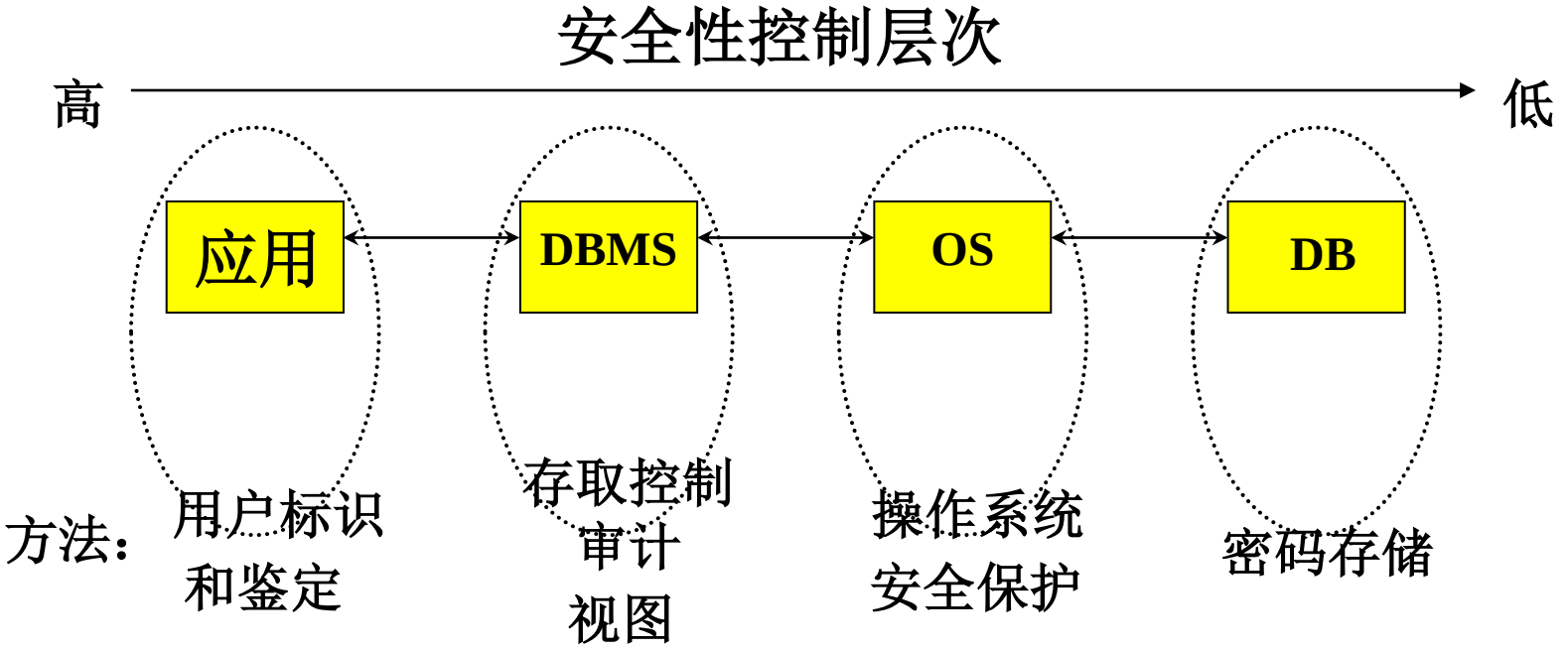
数据库安全性控制概述（续）

- 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

数据库安全性控制概述（续）



计算机系统的安全模型

数据库安全性控制概述（续）

- 数据库安全性控制的常用方法
  - 用户标识和鉴定
  - 存取控制
  - 视图
  - 审计
  - 密码存储

4.2 数据库安全性控制

4.2.1 用户标识与鉴别

4.2.2 存取控制

4.2.3 自主存取控制方法

4.2.4 授权与回收

4.2.5 数据库角色

4.2.6 强制存取控制方法

4.2.1 用户标识与鉴别

- 用户标识与鉴别

  （Identification & Authentication）

  - 系统提供的最外层安全保护措施

4.2.1 用户标识与鉴别

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

4.2.1 用户标识与鉴别

- 用户名/口令
  - 简单易行，容易被人窃取
- 每个用户预先约定好一个计算过程或者函数
  - 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定用户身份

4.2 数据库安全性控制

4.2.1 用户标识与鉴别

4.2.2 存取控制

4.2.3 自主存取控制方法

4.2.4 授权与回收

4.2.5 数据库角色

4.2.6 强制存取控制方法

4.2.2 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

存取控制（续）

- 定义存取权限
  - 在数据库系统中，为了保证用户只能访问他有权存取的数据，必须预先对每个用户定义存取权限。
- 检查存取权限
  - 对于通过鉴定获得上机权的用户（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。

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

自主存取控制方法

- 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

强制存取控制方法

- 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 数据库安全性控制

4.2.1 用户标识与鉴别

4.2.2 存取控制

4.2.3 自主存取控制方法

4.2.4 授权与回收

4.2.5 数据库角色

4.2.6 强制存取控制方法

4.2.3 自主存取控制方法

- 通过 SQL 的 GRANT 语句和 REVOKE 语句实现
- 用户权限组成
  - 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）

- 关系数据库系统中存取控制对象

对象类型	对象	操 作 类 型
数据库	模式	CREATE SCHEMA
	基本表	CREATE TABLE，ALTER TABLE
模式	视图	CREATE VIEW
	索引	CREATE INDEX
数据	基本表和视图	SELECT，INSERT，UPDATE，DELETE，REFERENCES，ALL PRIVILEGES
数据	属性列	SELECT，INSERT，UPDATE， REFERENCES
		ALL PRIVILEGES

自主存取控制方法（续）

- 检查存取权限
  - 对于获得上机权后又进一步发出存取数据库操作的用户
    - DBMS查找数据字典，根据其存取权限对操作的合法性进行检查
    - 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

自主存取控制方法（续）

授权粒度

- 授权粒度是指可以定义的数据对象的范围
  - 它是衡量授权机制是否灵活的一个重要指标。
  - 授权定义中数据对象的粒度越细，即可以定义的数据对象的范围越小，授权子系统就越灵活。

自主存取控制方法（续）

- 关系数据库中授权的数据对象粒度
  - 数据库
  - 表
  - 属性列
  - 行
- 能否提供与数据值有关的授权反映了授权子系统精巧程度

自主存取控制方法（续）

- 实现与数据值有关的授权
  - 利用存取谓词
    - 存取谓词可以很复杂

      可以引用系统变量，如终端设备号，系统时钟等，实现与时间地点有关的存取权限，这样用户只能在某段时间内，某台终端上存取有关数据

例：规定“教师只能在每年1月份和7月份星期一至星期五上午8点到下午5点处理学生成绩数据”。

4.2 数据库安全性控制

4.2.1 用户标识与鉴别

4.2.2 存取控制

4.2.3 自主存取控制方法

4.2.4 授权与回收

4.2.5 数据库角色

4.2.6 强制存取控制方法

4.2.4 授权与回收

一、GRANT

- GRANT语句的一般格式：

  GRANT <权限>[,<权限>]...

  [ON <对象类型> <对象名>]

  TO <用户>[,<用户>]...

  [WITH GRANT OPTION];

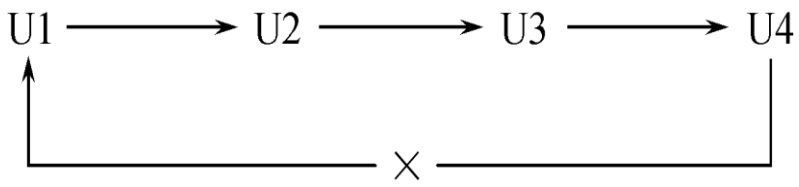
- 语义：将对指定操作对象的指定操作权限授予指定的用户

GRANT（续）

- 发出GRANT：
  - DBA
  - 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（全体用户）

WITH GRANT OPTION子句

- 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



例题

［例1］把查询Student表权限授给用户U1

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

例题（续）

[例2] 把对Student表和Course表的全部权限授予用户U2和U3

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

例题（续）

[例3] 把对表SC的查询权限授予所有用户

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

例题（续）

[例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

例题（续）

[例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5后，U5不仅拥有了对表SC的INSERT权限，还可以传播此权限：

[例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7] GRANT INSERT ON TABLE SC TO U7;

但U7不能再传播此权限。

传播权限（续）

下表是执行了［例1］到［例7］的语句后，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

授权与回收（续）

二、REVOKE

- 授予的权限可以由DBA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>]

  FROM <用户>[,<用户>]...;

REVOKE（续）

[例8] 把用户U4修改学生学号的权限收回

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

REVOKE（续）

[例9] 收回所有用户对表SC的查询权限

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

REVOKE（续）

[例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权限收回的时候必须级联（CASCADE）收回
- 系统只收回直接或间接从U5处获得的权限

REVOKE（续）

执行［例8］到［例10］的语句后，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

小结:SQL灵活的授权机制

- DBA：拥有所有对象的所有权限
  - 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
  - GRANT：授予其他用户
- 被授权的用户
  - “继续授权”许可：再授予
- 所有授予出去的权力在必要时又都可用REVOKE语句收回

授权与回收（续）

三、创建数据库模式的权限

- DBA在创建用户时实现

- CREATE USER语句格式

CREATE USER <username>

［WITH］［DBA | RESOURCE | CONNECT］

授权与回收（续）

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	可以	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 数据库安全性控制

4.2.1 用户标识与鉴别

4.2.2 存取控制

4.2.3 自主存取控制方法

4.2.4 授权与回收

4.2.5 数据库角色

4.2.6 强制存取控制方法

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

数据库角色

- 一、角色的创建

  CREATE ROLE <角色名>

- 二、给角色授权

  GRANT <权限>［，<权限>］…

  ON <对象类型>对象名

  TO <角色>［，<角色>］…

数据库角色

- **三、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］

- **四、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>］…

数据库角色（续）

[例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；
```

数据库角色（续）

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；
```

数据库角色（续）

[例12]　角色的权限修改

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

数据库角色（续）

[例13]

REVOKE SELECT

ON TABLE Student

FROM R1；

4.2 数据库安全性控制

4.2.1 用户标识与鉴别

4.2.2 存取控制

4.2.3 自主存取控制方法

4.2.4 授权与回收

4.2.5 数据库角色

4.2.6 强制存取控制方法

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

4.2.6 强制存取控制方法

- 强制存取控制（MAC)
  - 保证更高程度的安全性
  - 用户能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
    - 军事部门
    - 政府部门

强制存取控制方法（续）

- 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（Label）
  - 绝密（Top Secret）
  - 机密（Secret）
  - 可信（Confidential）
  - 公开（Public）
- 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）

强制存取控制方法（续）

- 强制存取控制规则

(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）

- 规则的共同点

  禁止了拥有高许可证级别的主体更新低密级的数据对象

强制存取控制方法（续）

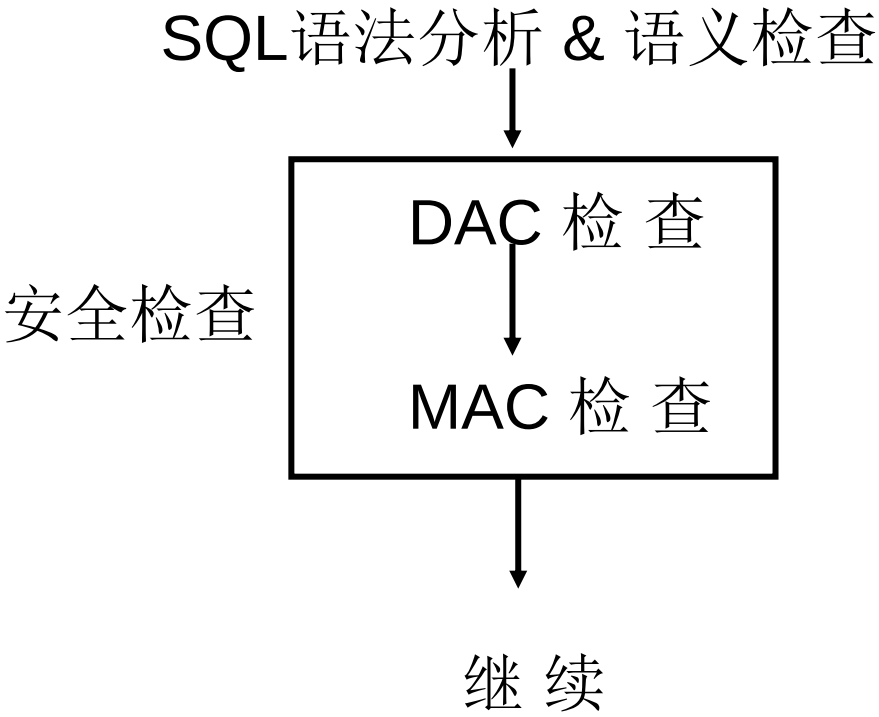
- 强制存取控制的特点
  - MAC是对数据本身进行密级标记
  - 无论数据如何复制，标记与数据是一个不可分的整体
  - 只有符合密级标记要求的用户才可以操纵数据
  - 从而提供了更高级别的安全性

MAC与DAC

- DAC与MAC共同构成DBMS的安全机制
- 实现MAC时要首先实现DAC
  - 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

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。

第四章 数据库安全性

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，
- 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。

4.3 视图机制

- 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14]建立计算机系学生的视图，把对该视图的SELECT权限授于王平，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';
```

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 PRIVILEGES
ON  CS_Student
TO  张明；
```

4.2 数据库安全性控制

4.1 计算机安全性概述

4.2 数据库安全性控制

4.3 视图机制

4.4 审计（Audit）

4.5 数据加密

4.6 统计数据库安全性

4.7 小结

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记录在上面

- DBA利用审计日志中的追踪信息

  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（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

审计（续）

- 审计分为
  - 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

审计（续）

- 审计分为
  - 系统级审计
    - 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

审计（续）

- AUDIT语句：设置审计功能

- NOAUDIT语句：取消审计功能

审计（续）

［例15］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16］取消对SC表的一切审计

NOAUDIT ALTER，UPDATE

ON SC；

4.2 数据库安全性控制

4.1 计算机安全性概述

4.2 数据库安全性控制

4.3 视图机制

4.4 审计（Audit）

4.5 数据加密

4.6 统计数据库安全性

4.7 小结

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）
  - 不知道解密算法的人无法获知数据的内容

4.5 数据加密

- 加密方法
  - 替换方法
    - 使用密钥（Encryption Key）将明文中的每一个字符转换为密文中的一个字符
  - 置换方法
    - 将明文的字符按不同的顺序重新排列
  - 混合方法

    美国1977年制定的官方加密标准：数据加密标准（Data Encryption Standard，简称DES）

4.5 数据加密

- DBMS中的数据加密
  - 有些数据库产品提供了数据加密例行程序
  - 有些数据库产品本身未提供加密程序，但提供了接口

4.5 数据加密

- 数据加密功能通常也作为可选特征，允许用户自由选择
  - 数据加密与解密是比较费时的操作
  - 数据加密与解密程序会占用大量系统资源
  - 应该只对高度机密的数据加密

第四章 数据库安全性

4.1 计算机安全性概述

4.2 数据库安全性控制

4.3 视图机制

4.4 审计（Audit）

4.5 数据加密

4.6 统计数据库安全性

4.7 小结

4.6 统计数据库安全性

- 统计数据库的特点
  - 允许用户查询聚集类型的信息（例如合计、平均值等）
  - 不允许查询单个记录信息

例：允许查询“程序员的平均工资是多少？”
不允许查询“程序员张勇的工资？”

4.6 统计数据库安全性

- 统计数据库中特殊的安全性问题
  - 隐蔽的信息通道
  - 从合法的查询中推导出不合法的信息

4.6 统计数据库安全性

例1：下面两个查询都是合法的：

1．本公司共有多少女高级程序员？

2．本公司女高级程序员的工资总额是多少？

如果第一个查询的结果是“1”，

那么第二个查询的结果显然就是这个程序员的工资数。

规则1：任何查询至少要涉及N(N足够大)个以上的记录

4.6 统计数据库安全性

例2：用户A发出下面两个合法查询：

1．用户A和其他N个程序员的工资总额是多少？

2．用户B和其他N个程序员的工资总额是多少？

若第一个查询的结果是X，第二个查询的结果是Y，

由于用户A知道自己的工资是Z，

那么他可以计算出用户B的工资=Y-(X-Z)。

Z+N=X

B+N=Y　B= Y-(X-Z)

原因：两个查询之间有很多重复的数据项

规则2：任意两个查询的相交数据项不能超过M个

4.6 统计数据库安全性

可以证明，在上述两条规定下，如果想获知用户B的工资额

A至少需要进行1+(N-2)/M次查询

规则3：任一用户的查询次数不能超过1+(N-2)/M

如果两个用户合作查询就可以使这一规定失效

4.6 统计数据库安全性

规则1：任何查询至少要涉及N(N足够大)个以上的记录

规则2：任意两个查询的相交数据项不能超过M个

规则3：任一用户的查询次数不能超过1+(N-2)/M

4.6 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

第四章 数据库安全性

4.1 计算机安全性概述

4.2 数据库安全性控制

4.3 视图机制

4.4 审计（Audit）

4.5 数据加密

4.6 统计数据库安全性

4.7 小结

4.7 小结

- 数据的共享日益加强，数据的安全保密越来越重要
- DBMS是管理数据的核心，因而其自身必须具有一整套完整而有效的安全性机制
- TCSEC和CC

小结（续）

- 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
- 自主存取控制功能
  - 通过SQL 的GRANT语句和REVOKE语句实现
- 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

下课了。。。

认真

休息一会儿。。。