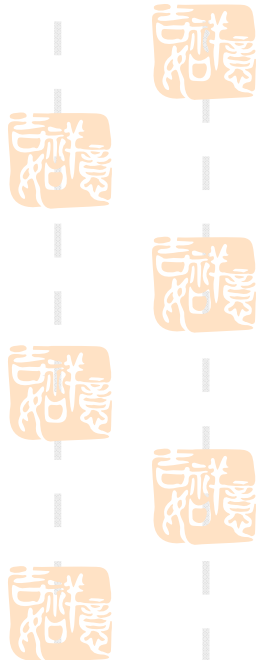




数据库系统概论

An Introduction to Database System

第四章 数据库安全性



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（续）

- 数据库的安全性是指保护数据库以防止不合法使用所造成的数据泄露、更改或破坏。
- 系统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。

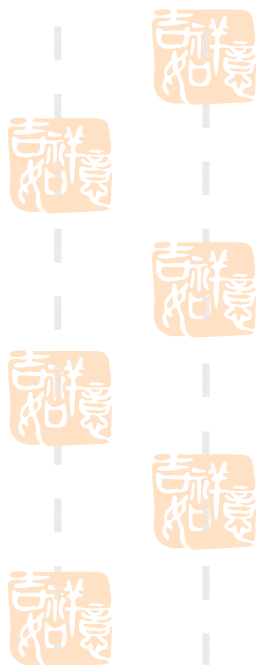


4.1 数据库安全性概述



4.1.1 数据库的不安全因素

4.1.2 安全标准简介



4.1.1 数据库的不安全因素

1. 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。



2. 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





3.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

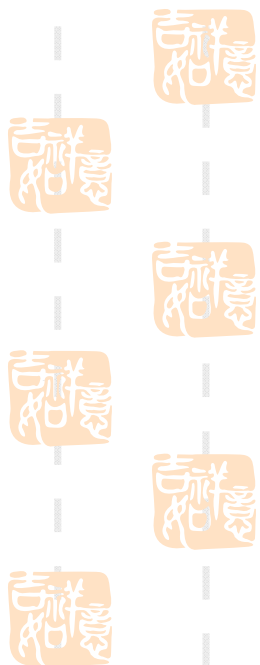
➤ 建立一套可信（Trusted）计算机系统的概念和
标准

4.1 数据库安全性概述



4.1.1 数据库的不安全因素

4.1.2 安全标准简介



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准则》（简称TCSEC或DoD85）
- 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）

安全标准简介（续）

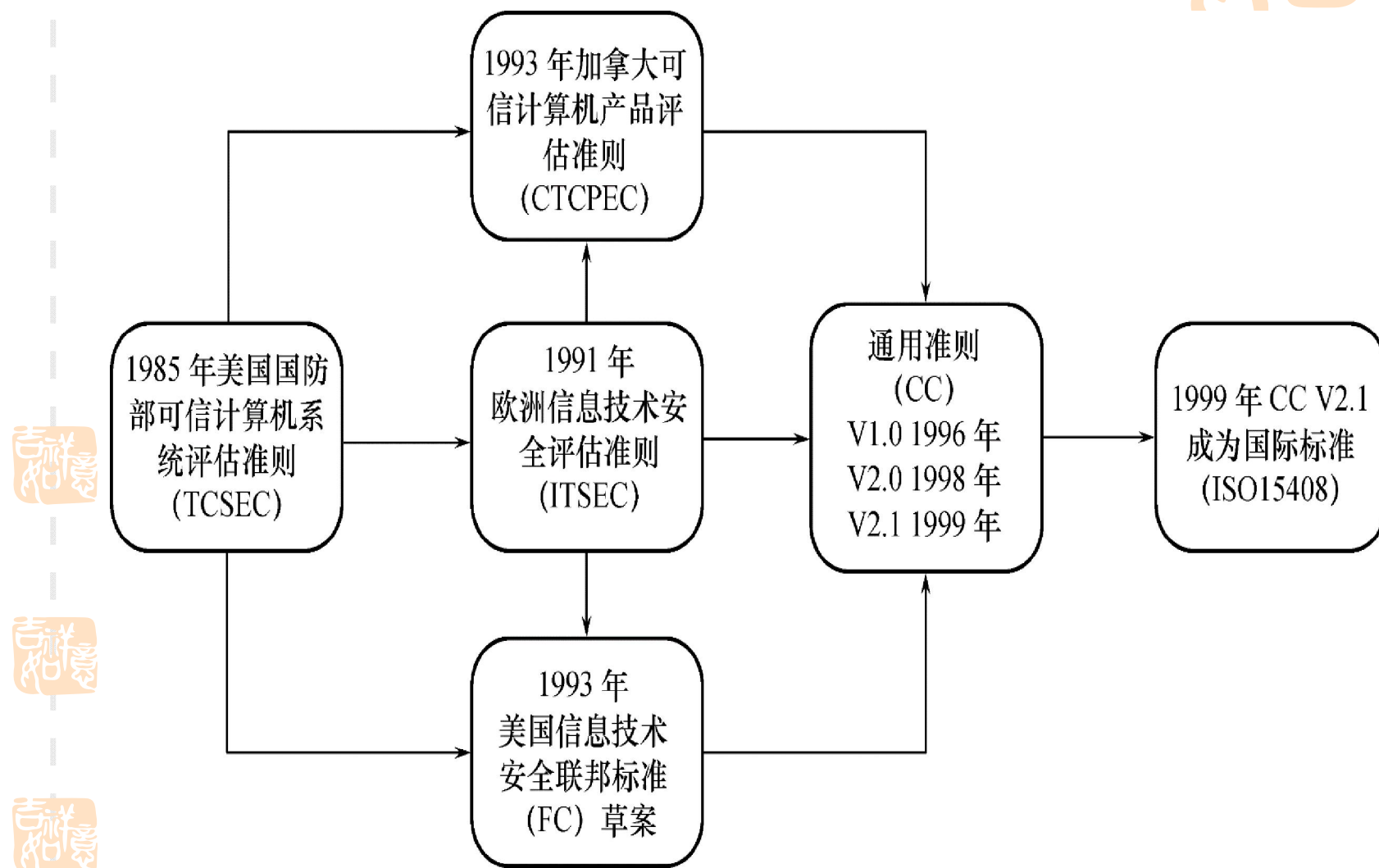
- 1993年，CTCPEC、FC、TCSEC和ITSEC联合行动，解决原标准中概念和技术上的差异，称为CC（Common Criteria）项目

- 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。

信息安全标准发展历史



4.1.2 安全标准简介



- TCSEC标准

- CC标准



TCSEC标准

- ① 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

② 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 ），提供高级别手段
B3	安全域（ Security Domains ），数据隐藏与分层、屏蔽
B2	结构化保护（ Structural Protection ），支持硬件保护
B1	标记安全保护（ Labeled Security Protection ）
C2	受控的存取保护（ Controlled Access Protection ），区分用户
C1	自主安全保护（ Discretionary Security Protection ），不区分用户
D	最小保护（ Minimal Protection ），几乎没有安全性可言

按系统可靠或可信程度逐渐增高
各安全级别之间：偏序向下兼容

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。
- 现有的商业系统稍作改进即可满足

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
- 典型例子
 - Windows 2000
 - Oracle 7

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
- B1级典型例子
 - 操作系统
 - 惠普公司的HP-UX BLS release 9.09+
 - 数据库
 - Oracle公司的Trusted Oracle 7
 - Sybase公司的Secure SQL Server version 11.0.6

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**

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

■ 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

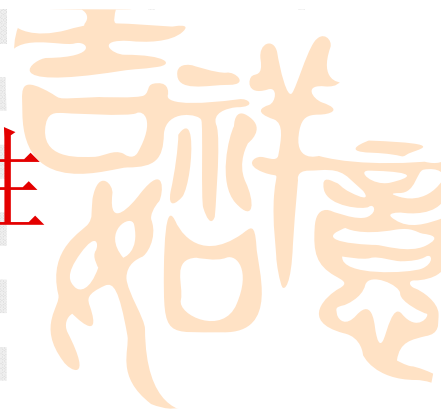


CC (续)

■ 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

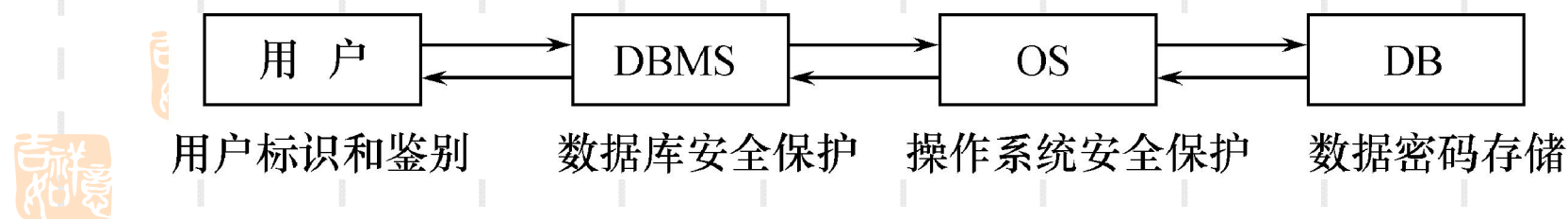
4.6 统计数据库安全性

4.7 小结



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

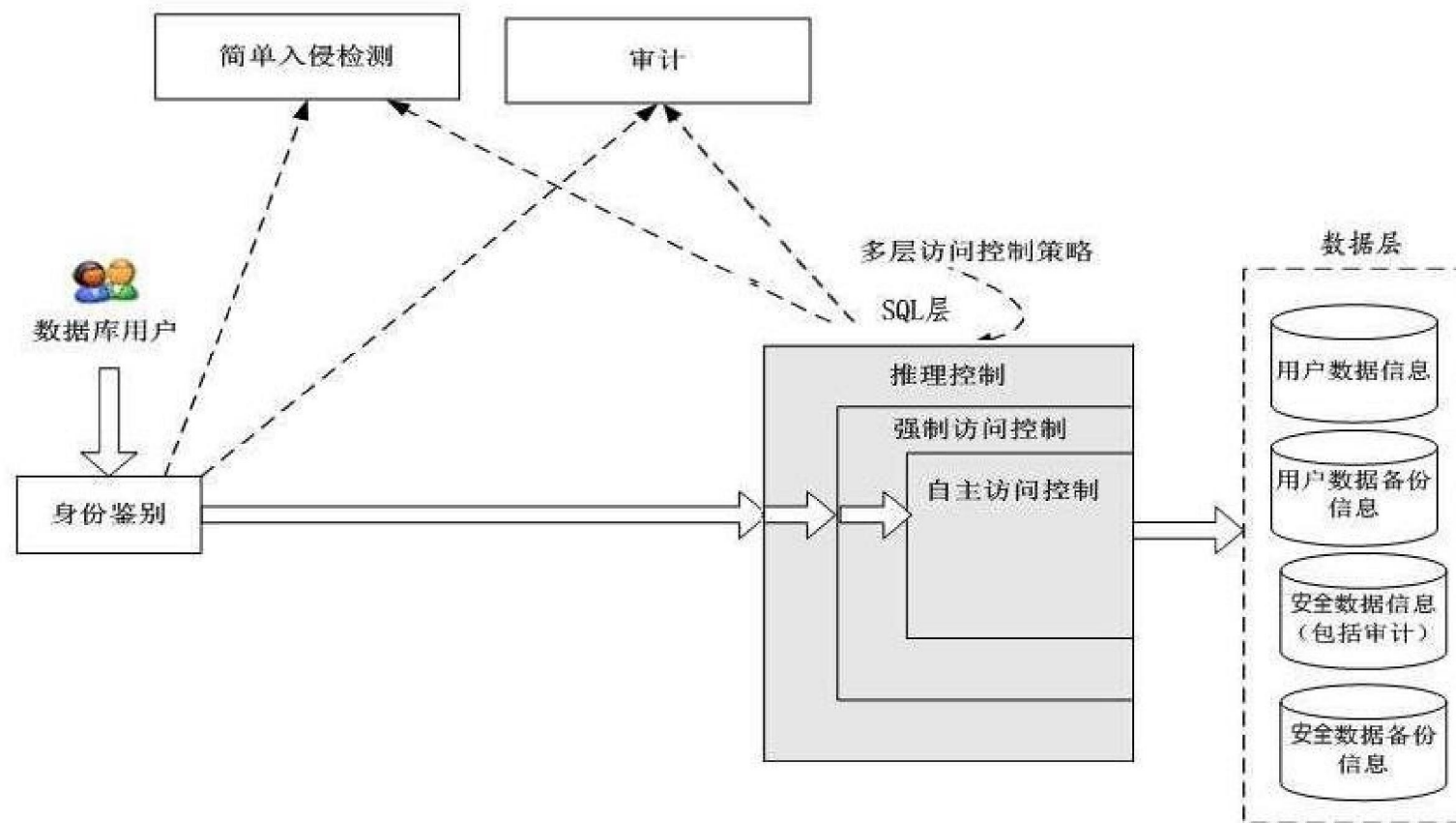


计算机系统的安全模型

数据库安全性控制（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



❖ 存取控制流程

- 首先，数据库管理系统对提出**SQL**访问请求的数据库用户进行身份鉴别，防止不可信用户使用系统。

- 然后，在**SQL**处理层进行自主存取控制和强制存取控制，进一步可以进行推理控制。

- 还可以对用户访问行为和系统关键操作进行审计，对异常用户行为进行简单入侵检测。



数据库安全性控制概述（续）

- 数据库安全性控制的常用方法

- 用户标识和鉴别

- 存取控制

- 视图

- 审计

- 加密存储

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



■ 口令



➤ 系统核对口令以鉴别用户身份



用户身份鉴别（续）



■ 用户身份鉴别的方法

1. 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

2. 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

3. 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

4. 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

4.2.2 存取控制

- 存取控制机制组成
 - 定义用户权限，并将用户权限登记到数据字典中
 - 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，存放在数据字典中，称做安全规则或授权规则
 - 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，进行合法权限检查
- 用户权限定义和合法权检查机制一起组成了数据库管理系统的存取控制子系统

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
- 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

存取控制（续）

■ 常用存取控制方法（续）

➤ 强制存取控制（Mandatory Access Control, 简称 MAC）

- B1级

- 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**

关系数据库系统中的权限

对象类型	对象	操作类型	
数据库 模式	模式	CREATE SCHEMA	语句权限(SQL Server)/系统权限(Oracle)
	基本表	CREATE TABLE , ALTER TABLE	
	视图	CREATE VIEW	
	索引	CREATE INDEX	
数据	基本表和视图	SELECT , INSERT , UPDATE , DELETE , REFERENCES , ALL PRIVILEGES	对象权限
	属性列	SELECT , INSERT , UPDATE , REFERENCES , ALL PRIVILEGES	

授权



- 授权：给予用户一定的访问特权
- 在SQL中，有两种授权
 - 授予某类数据库用户的特权，只能由DBA授予
 - 授予对某些数据对象进行某些操作的特权，可由DBA授予，也可由数据对象的创建者授予

■ 权限种类

➤ 对象权限

是指用户对数据库中的表、视图等对象中数据的操作权。

➤ 语句权限(SQL Server)/系统权限(Oracle)

相当于数据定义语言（DDL）的语句权限，这种权限专指是否允许执行：**CREATE TABLE**、**CREATE VIEW**等与创建数据库对象有关的操作。



第1种授权（对象权限）

授权：

GRANT <特权> **ON** 〈表名〉 **TO** <受权者>[, <受权者>]
[**WITH GRANT OPTION**]

特权: **ALL PRIVILEGES** |操作

操作: **SELECT** |**INSERT**|**DELETE**|**UPDATE** [属性表]

受权者: **PUBLIC** |用户标识符

收回特权：

REVOKE <特权> **ON** 〈表名〉 **FROM** <受权者>[, <受权者>]
>]

GRANT



➤ 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）

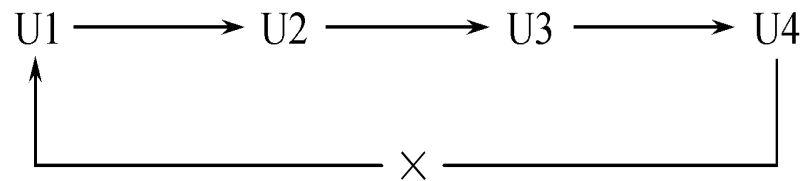


WITH GRANT OPTION子句

- 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 PRIVILIGES  
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

授权的例子



- DBA:

GRANT select, insert ON students TO CS3
WITH GRANT OPTION



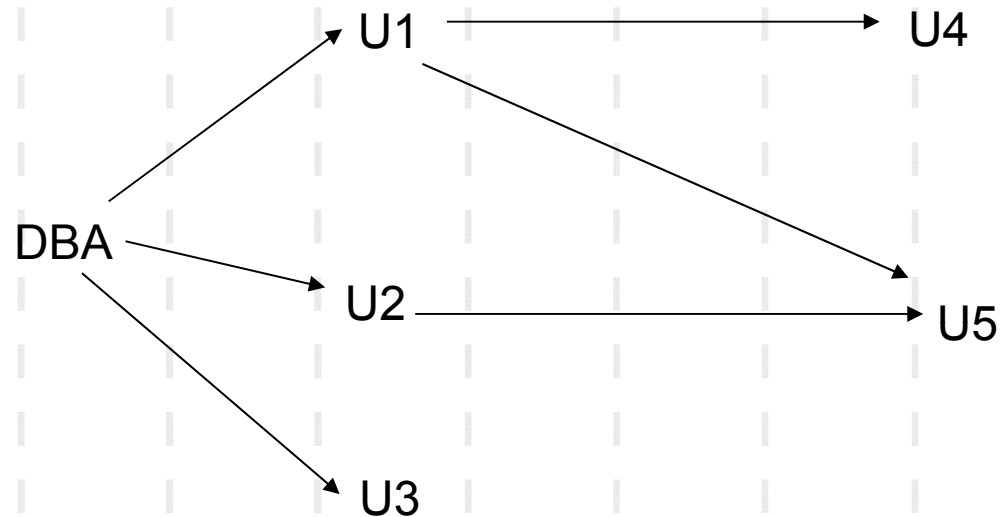
- CS3:

GRANT select ON students TO li4



权限转授

- 采用授权图表示权限的传递。
- 使用[with grant option]



- DBA:
Grant update on branch to U1,U2,U3 with grant option
- U1:
Grant update on branch to U4,U5
- U2:
Grant update on branch to U5

REVOKE

[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;



第2种授权



GRANT 权限名 [, ...]

TO {数据库用户名 | 用户角色名} [, ...]

收权语句

REVOKE 权限名 [, ...]

FROM { 数据库用户名 | 用户角色名 }
[, ...]



定义用户权限和授权的例子

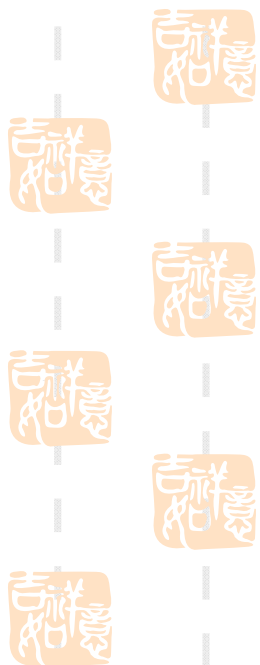
- 授予**user1**具有创建数据库表的权限。
 - **GRANT CREATE TABLE TO user1**
- 授予**user1**和**user2**具有创建数据库表和视图的权限。
 - **GRANT CREATE TABLE, CREATE VIEW TO user1, user2**



收回授权的命令和例子

例子:

REVOKE CREATE TABLE FROM user1



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

数据库用户类型



- 具有**CONNECT**权限的用户，只能连接数据库，成为一个数据库用户；这个用户最初除了连接，无其它任何权限，必须通过其它授权操作来增加权限。
- 具有**RESOURCE**权限的用户，可以创建表和索引，并可以控制这些资源的授权；
- 具有**DBA**特权的用户，有任何权限。



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

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

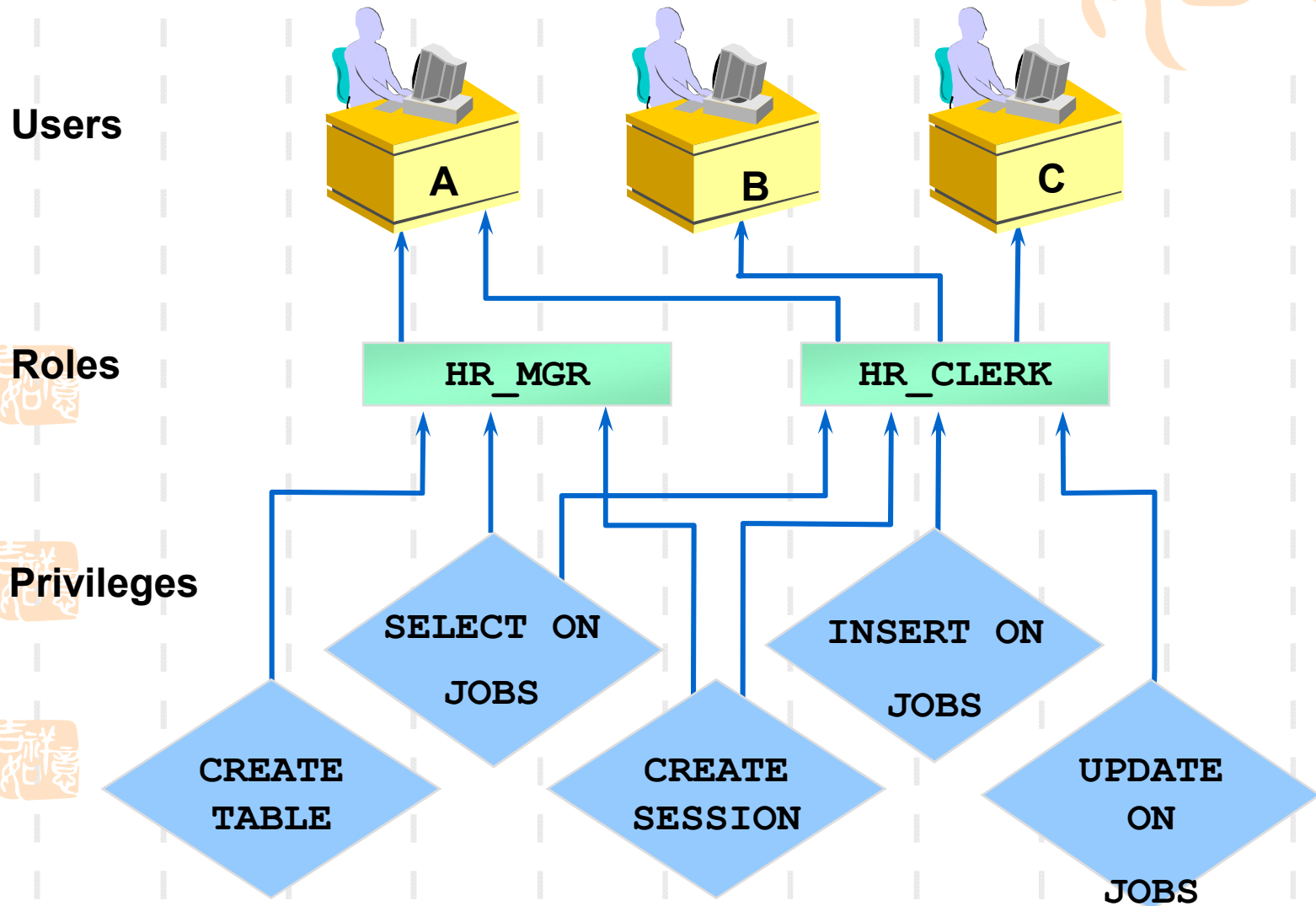
➤ 角色是权限的集合

➤ 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



角色(Roles)



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. 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；

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）

- 保证更高层次的安全性

- 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）

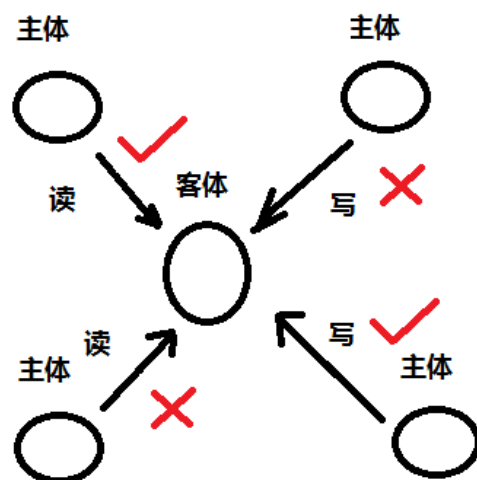
■ 客体的敏感度标记称为密级（Classification Level）

强制存取控制方法（续）

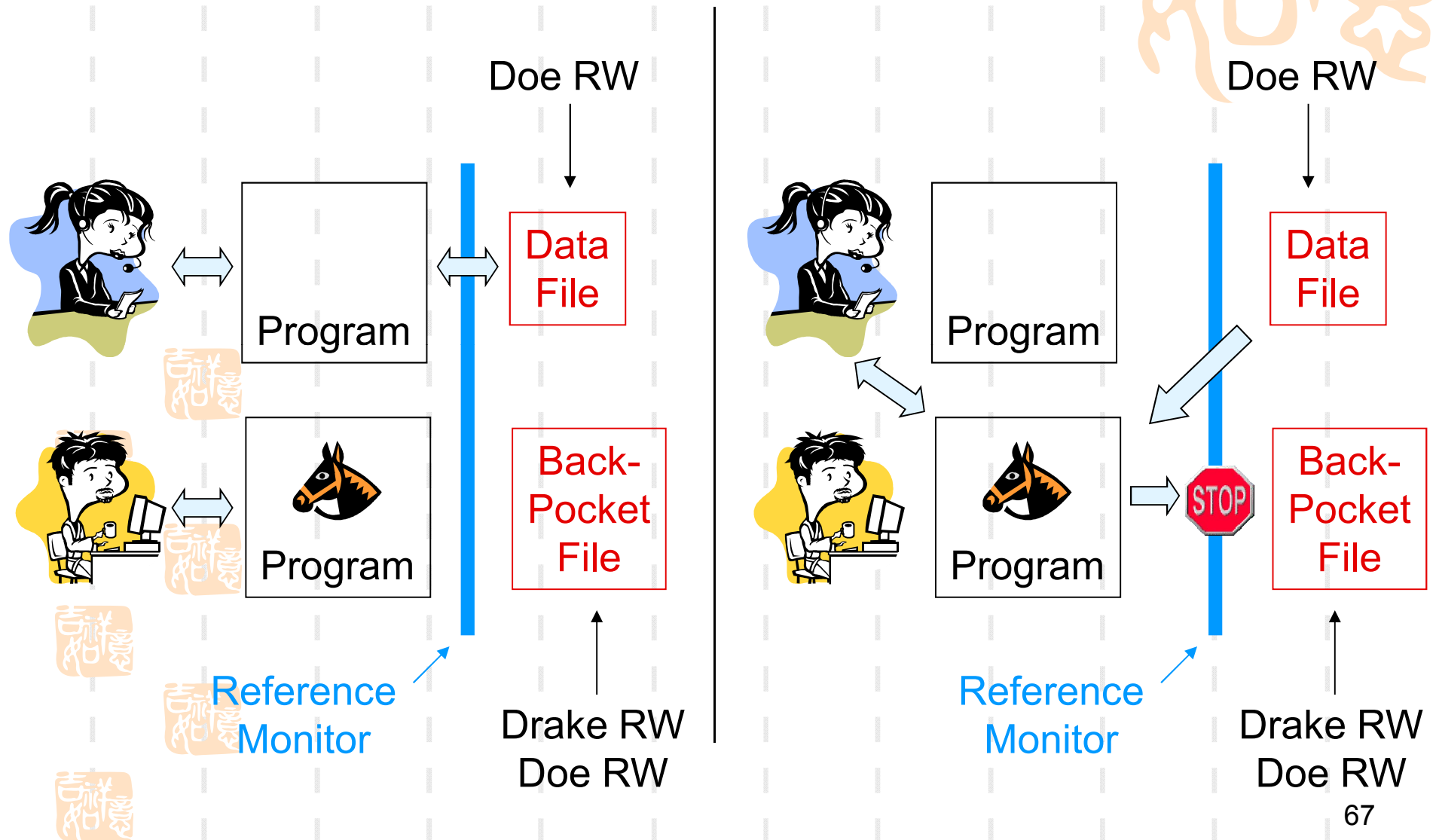
■ 强制存取控制规则

(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（**不往上读**）

(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（**不往下写**）



Trojan Horse – Flow Control



多级安全性的多级关系

EMPLOYEE

Name		Salary		JobPerformance		TC
Smith	U	40000	C	Fair	S	S
Brown	C	80000	S	Good	C	S

(a) 初始EMPLOYEE元组

EMPLOYEE

Name		Salary		JobPerformance		TC
Smith	U	40000	C	null	C	C
Brown	C	null	C	Good	C	C

(b) 为许可证级别为C的用户过滤后得到的EMPLOYEE关系

EMPLOYEE

Name		Salary		JobPerformance		TC
Smith	U	null	U	null	U	U

(c) 为许可证级别为U的用户过滤后得到的EMPLOYEE关系

EMPLOYEE

Name		Salary		JobPerformance		TC
Smith	U	40000	C	Fair	S	S
Smith	U	40000	C	Excellent	C	C
Brown	C	80000	S	Good	C	S

(d) Smith元组的多重实例

强制存取控制方法

- 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

► 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

■ 自主存取控制与强制存取控制共同构成数据库管理系统的安全机制

DAC + MAC安全检查

SQL语法分析 & 语义检查

安全检查

DAC 检 查

MAC 检 查

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。

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**)

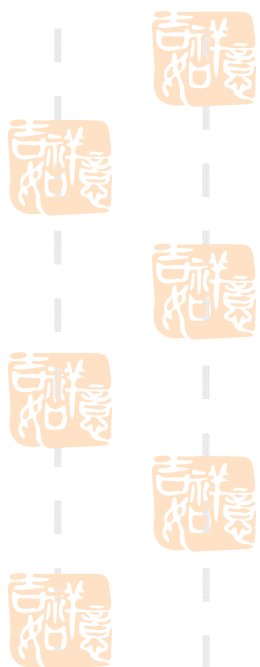
4.5 数据加密

4.6 统计数据库安全性

4.7 小结

4.3 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（续）

[例14]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;
```



4.2 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

➤ 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（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



审计事件



■ 服务器事件

- 审计数据库服务器发生的事件

■ 系统权限

- 对系统拥有的结构或模式对象进行操作的审计
- 要求该操作的权限是通过系统权限获得的



■ 语句事件

- 对SQL语句，如DDL、DML、DQL及DCL语句的审计



■ 模式对象事件

- 对特定模式对象上进行的SELECT或DML操作的审计



审计功能



- 基本功能

- 提供多种审计查阅方式提供多种审计查阅方式

- 多套审计规则：一般在初始化设定

- 提供审计分析和报表功能

- 审计日志管理功能

- 防止审计员误删审计记录，审计日志必须先转储后删除

- 对转储的审计记录文件提供完整性和保密性保护

- 只允许审计员查阅和转储审计记录，不允许任何用户新增和修改审计记录等

- 提供查询审计设置及审计记录信息的专门视图



审计设置和取消



- **AUDIT**语句：设置审计功能

- **NOAUDIT**语句：取消审计功能



审计分类



➤ 用户级审计

- 任何用户可设置的审计
- 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



➤ 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



审计例子



[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——明文（Plain text）变换为不可直接识别的格式——密文（Cipher text）

- 加密方法

- 存储加密
- 传输加密

数据加密（续）



❖ 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

内核级加密方法：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

加密表列

- 在希望加密的列名后使用**ENCRYPTION**关键字加密表的数据

```
CREATE TABLE EMPLOYEES (  
  first_name VARCHAR2(30),  
  last_name VARCHAR2(30),  
  emp_id NUMBER (9),  
  salary NUMBER(6),  
  ssn NUMBER(9) ENCRYPT) ;
```

程序包加密



例子: 用**3DES** 加密字符串

```
dbms_obfuscation_toolkit.DESEncrypt(input_string => input_string,  
    key_string => key_string,  
    encrypted_string => encrypted_string );  
return encrypted_string;
```



用**3DES** 解密字符串

```
dbms_obfuscation_toolkit.DESDecrypt(input_string => input_string,  
    key_string => key_string,  
    decrypted_string => decrypted_string );  
return decrypted_string;
```



数据加密（续）

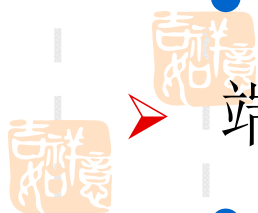
■ 传输加密

➤ 链路加密

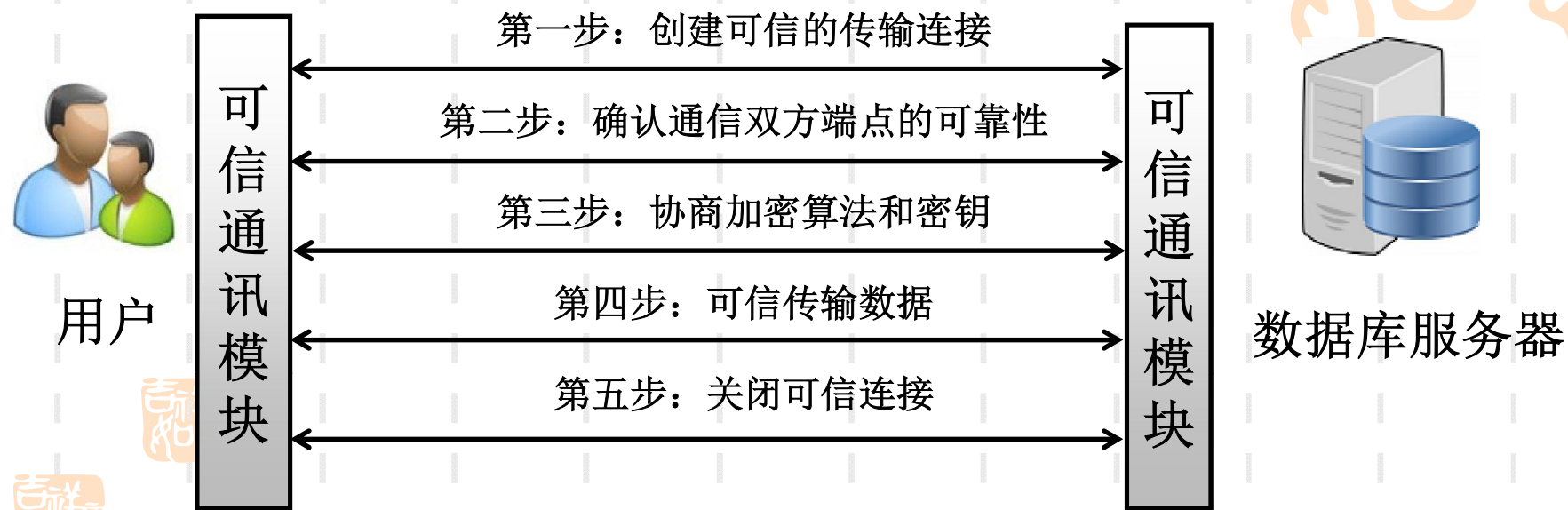
- 在链路层进行加密
- 传输信息由报头和报文两部分组成
- 报文和报头均加密

➤ 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



数据库管理系统可信传输示意图（端对端加密）



① 确认通信双方端点的可靠性

采用基于数字证书的服务器和客户端认证方式

通信时均首先向对方提供己方证书，然后使用本地的CA信任列表和证书撤销列表对接收到的对方证书进行验证

② 协商加密算法和密钥：确认双方端点的可靠性后，通信双方协商本次会话的加密算法与密钥

③ 可信数据传输：业务数据在被发送之前将被用某一组特定的密钥进行加密和消息摘要计算，以密文形式在网络上传输

当业务数据被接收的时候，需用相同一组特定的密钥进行解密和摘要计算

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

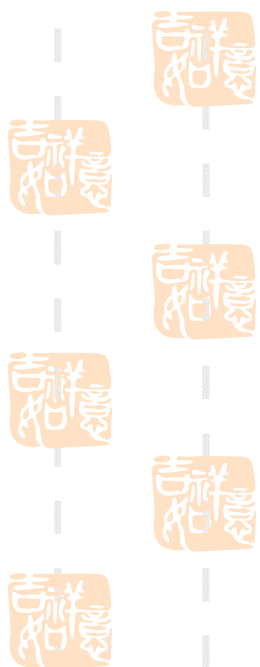
4.6 其他安全性保护

4.7 小结

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



推理控制

- 提供用户综合性数据为主的数据库称为统计数据库
- 允许访问综合性数据而禁止访问单个记录的详细机密性信息称为统计数据库安全性。相应的控制措施被称为推理控制（inference control）

- 统计数据库中特殊的安全性问题

➤ 从综合性数据推断单个记录的值

- 例子：Q1：本公司有多少女高级程序员？

```
select count(*) from emp where sex='女' and job='高级程序员'
```

Q2：本公司女高级程序员的工资总额是多少？

```
select sum(salary) from emp where sex='女' and job='高级程序员'
```

例子

- 你可以从病人数据库中查询统计数据,如计数,求和,求平均.
 - 查询: 多少个病人是男性、45岁-50岁、已婚、有两个孩子、银行副总裁?
 - **Answer:** 1
 - 查询:多少个病人是男性、45岁-50岁、已婚、有两个孩子、银行副总裁并且药物上瘾?
 - **Answer:** 1
- 如果你知道第1个查询的结果为1, 你可以查出这个人的所有信息.

防备策略



■ 查询控制 (Query controls)

- 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
- 规则3: 任一用户的查询次数不能超过 $1+(N-2)/M$

■ 条目控制 (Item controls)

- 抑制 (Suppression): 包含敏感数据的查询被拒绝
- 隐藏 (Concealing): 提供的答案是接近的但不是准确的.



防备策略(Cont.)

- 随机采样:
 - Result is computed on a random selected subset of the database but not the whole database.
- 随机数据污染:
 - Result is perturbed by a small error.
- 查询分析:
 - A query and its implications are analyzed.
 - Complexity:
 - Maintain a query history for each user.
 - Judge each query on the context of previous

- 没有一种完整的解决方案 – 所有的查询都带着某种信息
- 最后的防患 – 审计(**auditing**)
 - 检查不寻常的查询的审计记录

其他安全性保护（续）



- 隐蔽信道

- 处理强制存取控制未解决的问题

- 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 小结

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

小结（续）

- 实现数据库系统安全性的技术和方法
 - 用户身份鉴别
 - 存取控制技术：自主存取控制和强制存取控制
 - 视图技术
 - 审计技术
 - 数据加密存储和加密传输

小结（续）



- 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



作业

- 5
- $7(1), (2), (6), (7)$
- $8(\text{对应于 } 7(1), (2), (6), (7))$

