数据库系统概论

An Introduction to Database System

第四章　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）

- 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。
- 各系统安全性之间是相互紧密联系、相互支持的

第四章　数据库安全性

4.1　计算机安全性概论

4.2　数据库安全性控制

4.3　　视图机制

4.4　　审计

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

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

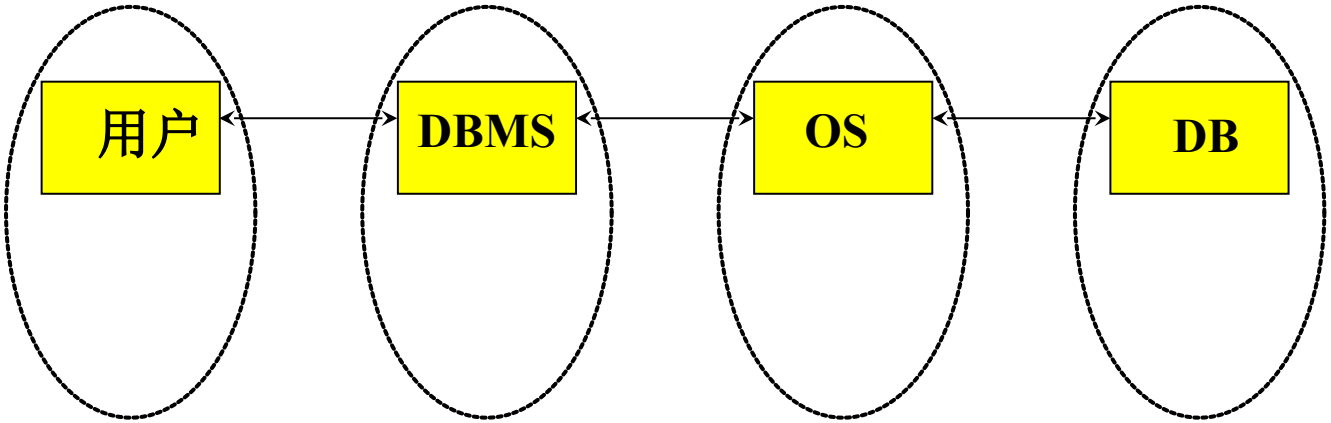
4.2.4 授权与回收

4.2.5 数据库角色

4.2.6 强制存取控制方法

计算机系统中的安全模型

安全性控制层次



用户标识和鉴定

数据库安全保护

操作系统安全保护

密码存储

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

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

用户标识自己的名字或身份

- 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

4.2.2 存取控制

- 存取控制机制的功能
  - 存取控制机制的组成
    - 定义存取权限
    - 检查存取权限

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

1. 自主存取控制（Discretionary Access Control ，简称 DAC ）

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

1. 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

2. 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

4.2.3 自主存取控制方法

- 优点
  - 能够通过授权机制有效地控制其他用户对敏感数据的存取
- 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

4.2.4 授权与回收

- 谁定义？

  DBA、表的建立者（即表的属主）或已经拥有该权限的用户。

- 如何定义？

  SQL 语句：

  GRANT 及 REVOKE

一 授 权 (GRANT)

- GRANT 语句的一般格式：

  GRANT <权限>[,<权限>]...

  [ON <对象名>] TO <用户>[,<用户>]...

  [WITH GRANT OPTION];

- **GRANT 功能：将对指定操作对象的指定操作权限授予指定的用户。**

(1) 用户的权限

- 建表（ CREATE TABLE ）的权限：属于DBA
- DBA 授予 --> 普通用户
- 基本表或视图的属主拥有对该表或视图的一切操作权限
- 接受权限的用户：

  一个或多个具体用户

  PUBLIC （全体用户）

(2) WITH GRANT OPTION 子句

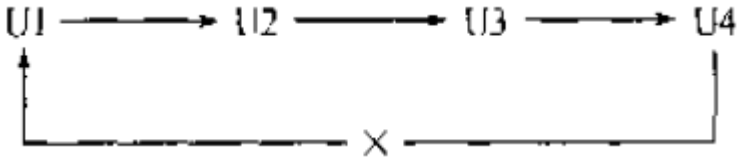
- 指定了 WITH GRANT OPTION 子句：

  获得某种权限的用户还可以把这种权限再授予别的用户。

- 没有指定 WITH GRANT OPTION 子句：

  获得某种权限的用户只能使用该权限，不能传播该权限

注：不允许循环授权，即被授权者不能把权限再授回给授权者或其祖先



例题

例 1　把查询 S 表权限授给用户 U1

```
GRANT   SELECT
ON      S
TO   U1;
```

例题（续）

例 2　把对 S 表和 C 表的全部权限授予用户 U2 和 U3

```
GRANT ALL PRIVILEGES
ON  S, C
TO U2, U3;
```

只能单表授权。

例题（续）

例 3　把对表 SC 的查询权限授予所有用户

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

例题（续）

例 4　把查询 S 表和修改学生学号的权限授给用户 U4

```
GRANT UPDATE(Sno), SELECT
ON  S
TO U4;
```

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 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 SC TO U6
WITH GRANT OPTION;

同样， U6 还可以将此权限授予 U7 ：

例 7 GRANT INSERT ON SC TO U7;

但 U7 不能再传播此权限。

U5--> U6--> U7

例题（续）

例 6 DBA 把在数据库 S_C 中建立表的权限授予用户 U8

```
GRANT CREATETAB
ON DATABASE S_C
TO U8;
```

使用：

```
GRANT CREATE TABLE
TO U8
```

sp_helprotect 系统存储过程可报告对数据库级安全对象的权限。

二 收回权限 (REVOKE)

- REVOKE 语句的一般格式为：

  REVOKE <权限>[,<权限>]...
  [ON <对象名>]
  FROM <用户>[,<用户>]...;

- 功能：从指定用户那里收回对指定对象的指定权限

例题

例 8　把用户 U4 修改学生学号的权限收回

```
REVOKE UPDATE(Sno)
ON S
FROM U4;
```

例题（续）

例 9　收回所有用户对表 SC 的查询权限

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

例题（续）

例 10　把用户 U5 对 SC 表的 INSERT 权限收回

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

例 11　权限的级联回收

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

权限的级联回收

系统将收回直接或间接从 U5 处获得的对 SC 表的 INSERT 权限：

-->U5--> U6--> U7

收回 U5 、 U6 、 U7 获得的对 SC 表的 INSERT 权限：

<--U5<-- U6<-- U7

4.2.5 数据库角色

- 数据库角色是被命名的一组与数据库操作相关的权限，是权限的集合

1) 角色的创建

- EXEC SP_ADDROLE < 角色名 >

2) 给角色授权

- GRANT ….. ON ….. TO < 角色名 >

3) 将一个角色授予其他的角色或用户

4) 角色权限的回收

REVOKE…..ON……FROM < 角色名 >

- 首先创建一个角色 R1

```
CREATE ROLE R1;
```

- 使用 GRANT 语句，使角色 R1 拥有 SC 表的若干权限

```
GRANT SELECT, UPDATE, INSERT
ON SC
TO R1;
```

- 将这个角色授予 R2，是它具有角色 R1 所包含的权限

```
GRANT R1
TO R2;
```

- 可以一次性通过 R1 来回收 R2 的对应权限

```
REVOKE R1
FROM R2 ;
```

4.2.6 强制存取控制方法

- 自主存取控制（DAC）：用户可以自由地决定将数据的存取权限授予何人、决定是否也将“授权”的权限授予别人。即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标记。
- 强制存取控制（MAC）：
  - 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

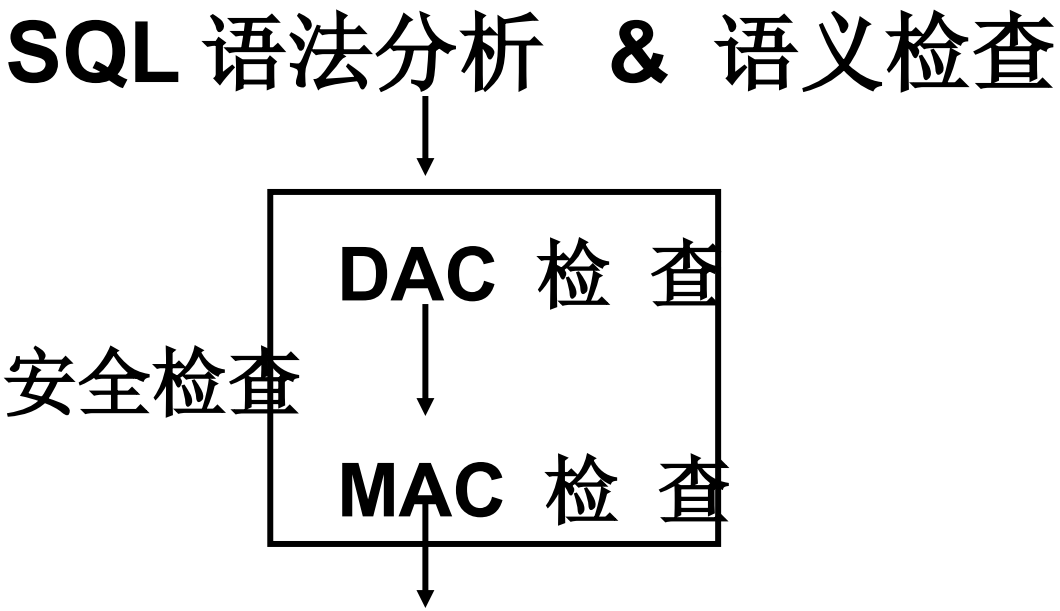
  提供了更高级别的安全性，适用于那些对数据有严格而固定密级分类的部门，例如军事部门或者政府部门。

MAC 与 DAC

- DAC （自主存取控制）与 MAC （强制存取控制）共同构成 DBMS 的安全机制
- 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。

强制存取控制方法（续）

DAC + MAC 安全检查示意图



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。

视图机制（续）

例：王平只能检索计算机系学生的信息

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 ；
```

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 可以利用审计日志中的追踪信息

    找出非法存取数据的人

4.4 审计

- 审计的分类
- **用户级审计：**任何用户可设计的审计，主要是用户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或不成功的访问要求及各种类型的 SQL 操作。
- **系统级审计：**只能有 DBA 设置，用以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）

- 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）

- AUDIT 语句用来设置审计功能：

AUDIT ALTER， UPDATE

ON SC；

- NOAUDIT 语句用来取消审计功能：

NOAUDIT ALTER， UPDATE

ON SC；

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

数据加密（续）

- 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 ）将替换方法和置换方法相结合，安全程度高。

数据加密（续）

- 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

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？”

统计数据库安全性（续）

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

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

统计数据库安全性（续）

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)。

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