数据库系统概论-04-数据库安全性

2022-06-16 | 2025-11-26 | 南京大学软件学院本科课程， 2022Spring-数据库系统概论 | 104

3.1k | 3 分钟

1. 自主存取控制

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

数据库管理员：

1. 拥有所有对象的所有权限
2. 根据实际情况不同的权限授予不同的用户

用户：

1. 拥有自己建立的对象的全部的操作权限
2. 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。

1.1 GRANT

发出 `GRANT` 的用户：

1. 数据库管理员数据库对象创建者（即属主 Owner）
2. 拥有该权限的用户

接受权限的用户：

1. 一个或多个具体用户
2. `PUBLIC`（即全体用户）

对象类型	对象	操作类型
数据库模式	模式	CREATE SCHEMA
	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
	属性列	SELECT，INSERT，UPDATE，REFERENCES，ALL PRIVILEGES

```
# GRANT 语句的一般格式：
# WITH GRANT OPTION 子句：可以再授予其他用户，不允许循环授权
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 ];

# [例 4.1] 把查询 Student 表权限授给用户 U1
GRANT SELECT ON TABLE Student
    TO U1;

# [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;

# [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;

# [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;

# [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;

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

# [例 4.7]同样，U6 还可以将此权限授予 U7，但 U7 不能再传播此权限。
```

```
GRANT INSERT ON TABLE SC
    TO U7;
```

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

1.2 REVOKE

```
# 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];

# [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;

# [例 4.9] 收回所有用户对表 SC 的查询权限
REVOKE SELECT ON TABLE SC
    FROM PUBLIC;

# [例 4.10] 把用户 U5 对 SC 表的 INSERT 权限收回
# 将用户 U5 的 INSERT 权限收回的时候使用 CASCADE，则同时收回 U6 或 U7 的 INSERT 权限，否则拒绝执行该语
# 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
REVOKE INSERT ON TABLE SC
    FROM U5 CASCADE;
```

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

1.3 创建数据库模式的权限

```
# 该命令不是 SQL 标准
CREATE USER <username> [WITH][DBA | RESOURCE | CONNECT];
```

只有系统的超级用户才有权创建一个新的数据库用户。

1. CONNECT 权限：如没有指定创建的新用户的权限，默认该用户拥有该权限。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。
2. RESOURCE 权限：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。
3. DBA 权限：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。

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

1.4 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>]...;

# 将一个角色授予其他的角色或用户
# 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
GRANT <角色 1>[,<角色 2>]… TO <角色或用户>[,<角色或用户>]... [WITH ADMIN OPTION];

# 角色权限的收回
# 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或拥有在这个（些）角色上的 ADMIN OPTION
REVOKE <权限>[,<权限>]... ON <对象类型> <对象名> FROM <角色>[,<角色>]...;

# [例 4.11] 通过角色来实现将一组权限授予一个用户。
CREATE ROLE R1;
GRANT SELECT, UPDATE, INSERT ON TABLE Student
    TO R1;
GRANT R1 TO 王平,张明,赵玲;
REVOKE R1 FROM 王平;

# [例 4.12] 角色的权限修改
GRANT DELETE ON TABLE Student
TO R1;

# [例 4.13] 使 R1 减少了 SELECT 权限
REVOKE SELECT ON TABLE Student
FROM R1;
```

2. 强制存取控制

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

1. 每一个数据对象被标以一定的密级
2. 每一个用户也被授予某一个级别的许可证
3. 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

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）。

敏感度标记分成若干级别

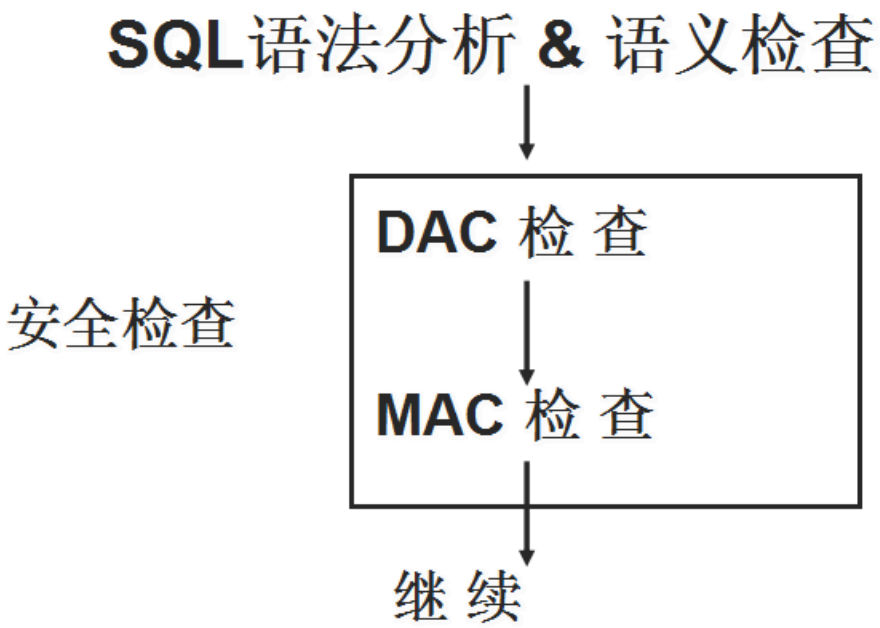
- 绝密（Top Secret，TS）

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

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小于或等于客体的密级时，该主体才能写相应的客体

> 规则 1 的必要性是明显的。
>
> 规则 2 是为了防止把数据的密级从高流向低。如许可证级别为 TS 的主体读取密级为 TS 的客体并以密级 P 写回，造成数据泄露。在规则 2 下，用户可以把写入的对象赋予高于自己许可证级别的密级，一旦数据被写入，该用户自己也不能读取该数据了。
>
> 规则 2 应该理解为：主体写入的客体内容**可以被赋予高于主体许可证级别的密级**，而不是：主体可以修改密级高于该主体许可证级别的客体。



打赏

原创

数据库系统概论-03-关系数据库标准语言 SQL
数据库系统概论-05-数据库完整性