安全性

- 安全性：防止恶意更新或偷窃数据的企图
- 数据库系统级
  - 验证和授权机制使得特定用户存取特定数据
  - 本章中主要讨论授权机制
- 操作系统级
  - 操作系统超级用户可对数据库做任何事情！ 需要有一个好的操作系统级安全机制
- 网络级：使用加密防止
  - 偷听(未授权的读取信息)
  - 伪装(冒充授权用户)

安全性

- 物理级
  - 对计算机的物理访问使得入侵者可摧毁数据，需要传统的锁钥安全手段
  - 防止洪水，火灾等对计算机的损坏
- 人员级
  - 审查用户以确保授权用户不会将存取权给予入侵者
  - 训练用户选择口令与保密

授权

- 对数据的授权包括：
  - 读权限 - 允许读，但不允许更新数据
  - 插入权限 - 允许插入新数据，但不允许更新现有数据
  - 修改权限 - 允许修改，但不允许删除数据
  - 删除权限 - 允许删除数据

授权

- 对修改数据库模式的授权包括：
  - 索引权限 – 允许创建和删除索引
  - 资源权限 – 允许创建新关系
  - 修改权限 – 允许增加或删除关系的属性
  - 删除权限 – 允许删除关系

授权与视图

- 用户可被授予关于视图的权限，而不被授予关于该视图定义中涉及的关系的权限
- 视图隐藏数据的能力既能简化系统的使用又能增强安全性(只允许用户存取他们工作中需要的数据)
- 关系级安全性与视图级安全性的结合使用可精确地将用户存取限制在他所需要的数据上

视图例

- 假设，一个工作人员只需要知道一个给定系（比如Geology系）里所有员工的工资，但无权看到其他系中员工的相关信息
  - 方法：不允许对*instructor*关系的直接访问，但授予对视图*geo_instructor*的访问权限，该视图仅由属于Geology系的那些*instructor*元组构成
  - 视图*geo_instructor*用SQL定义如下：

```
create view geo_instructor as
(select *
 from instructor
 where dept_name = ’ Geology’ );
```

视图例

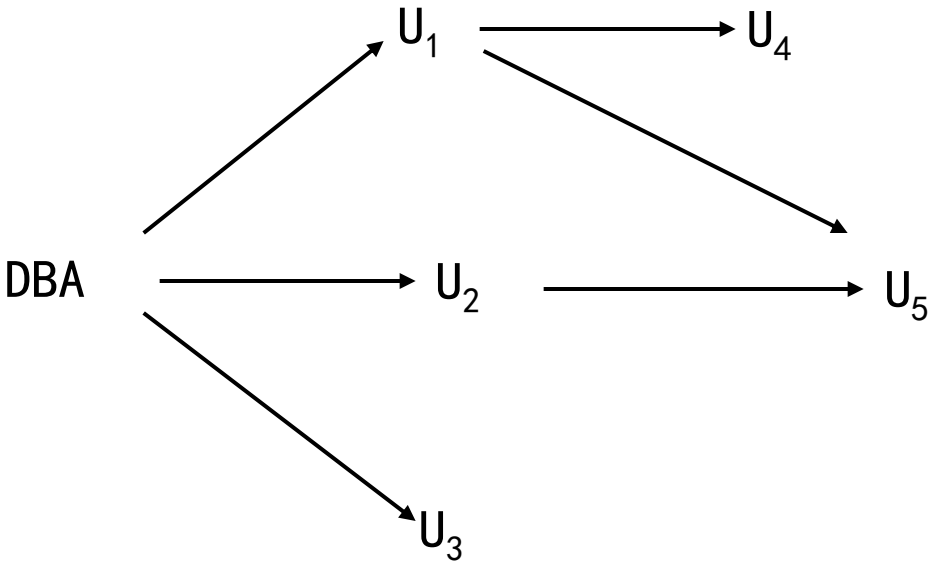
- 该工作人员有权看到如下查询的结果

```
select *
from geo_instructor;
```

- 当查询处理器将该查询转换为对数据库中实际关系的查询时，它产生了一个在*instructor*上的查询
- 必须在查询处理开始之前检查该工作人员的权限

权限的授予

- 权限从一个用户到另一个用户的传递可用授权图表示
- 图的节点是用户
- 图的根是数据库管理员
- 边$U_i \rightarrow U_j$表示用户U_i将某权限授予给了用户U_j



授权图

- 要求：授权图中的所有边都必须是某条从数据库管理员出发的路径的一部分
- 若DBA从U_1收回权限：
  - 必须从U_4收回权限，因为U_1不再有权限
  - 不能从U_5收回权限，因为U_5还有从DBA经U_2 的另一条授权路径
- 必须防止不经过根节点的循环授权：
  - DBA授权给U_7
  - U_7授权给U_8
  - U_8授权给U_7
  - DBA从U_7收回权限
- 必须收回从U_7到U_8以及从U_8到U_7的授权，因为不再有从DBA到U_7或U_8的路径

SQL中的安全性声明

- grant语句用于授权

  GRANT <*privilege list*>

  ON <*relation name or view name*> TO <*user list*>;

- < *user list* > 可以是：
  - 用户ID
  - public，代表所有合法用户
  - 角色
- 授予对视图的权限并不意味着授予对定义该视图的基础关系的权限
- 权限的授予者本身必须拥有相应的权限（或者是数据库管理员）

SQL中的权限

- select：允许读关系，或查询视图
  - 例如：授予用户U_1，U_2，U_3 对*instructor*关系的select权限：

    grant select on *instructor* to U_1 , U_2 , U_3
- insert ： 允许插入元组
- update ： 允许修改元组
- delete ： 允许删除元组
- references ： 创建关系时允许声明外键
- all privileges ： 所有权限

授权的权限

- with grant option：允许用户把被授予的权限再转授给其他用户
  - 例如：授予U_1对*instructor*的select权限并允许 U_1将此权限授予其他用户

    grant select on *instructor* to U_1 with grant option

角色

- 通过创建角色可以一次性对一类用户指定其共同的权限
- 像对用户一样，可以对角色授予或收回权限
- 角色可被赋予给用户，甚至给其他角色
- SQL:1999 支持角色

```
create role instructor;
grant select on takes to instructor;
grant dean to Amit;
create role dean;
grant instructor to dean;
grant dean to Satoshi;
```

SQL中的权限回收

- revoke语句用于回收权限

  REVOKE<*privilege list*> ON <*relation name or view name*>
  FROM <*user list*> [restrict | cascade]

- 例如:

  revoke select on *instructor* from U_1, U_2, U_3 cascade

- 从一用户收回权限可能导致其他用户也失去该权限，称为级联回收
- 指定restrict可以阻止级联回收

  revoke select on *instructor* from U_1, U_2, U_3 restrict

  - 如果要求级联回收，则带有restrict的revoke命令将会失败

SQL中的权限回收

- <privilege list> 可以是all ，以便收回某用户拥有的所有权限
- 如果同一权限被不同授予者两次授予同一用户，则该用户在回收一次后仍保持该权限
- 所有依赖于被收回权限的权限也被收回

SQL授权的局限性

- SQL不支持元组级的授权
  - 例如我们不能限制学生只能看他自己的分数
- 某些应用(如web应用)的所有最终用户可能被映射成单个数据库用户
- 以上情况下的授权任务只能依靠应用程序
  - 细粒度授权，如授权给个别元组，可以由应用程序来实现（优点）
  - 授权在应用程序代码中完成，并可能散布在整个应用中（缺点）
  - 检查是否有权限漏洞非常困难，因为需要读大量应用程序代码（缺点）

审计跟踪

- 审计跟踪（audit trail）是关于应用程序数据的所有更改（插入/删除/更新）的日志，以及一些信息，如哪个用户执行了更改和什么时候执行的更改
- 用于跟踪安全漏洞或错误更新
- 可以使用触发器实现审计跟踪，但是很多数据库提供了内置的机制创建审计跟踪

oracle中的审计

- 语句审计：

```
audit table by scott by access whenever successful;
```

  - 审计用户scott每次成功地执行有关table的语句(create table, drop table, alter table)
  - 格式：

```
AUDIT <st-opt> [ BY <users> ]
[ BY SESSION | ACCESS ]
[ WHENEVER SUCCESSFUL| WHENEVER NOT SUCCESSFUL ]
```

    - 当BY <users> 缺省，对所有用户审计
    - BY SESSION每次会话期间，相同类型的需审计的SQL语句仅记录一次
    - 常用的<st-opt>：table，view，role，index，…
    - 取消审计：NOAUDIT …(其余同audit语句)

oracle中的审计

- 对象（实体）审计：

```
audit delete, update on student;
```

  - 审计所有用户对student表的delete和update操作
  - 格式：

```
AUDIT <obj-opt> ON <obj>|DEFAULT
[ BY SESSION | BY ACCESS ]
[ WHENEVER SUCCESSFUL| WHENEVER NOT SUCCESSFUL ]
```

    - obj-opt: insert, delete, update, select, grant, …
    - 实体审计对所有的用户起作用
    - ON <obj> 指出审计对象表、视图名
    - ON DEFAULT 对其后创建的所有对象起作用
    - 取消审计：NOAUDIT …

oracle中的审计

- 怎样看审计结果：
  - 审计结果记录在数据字典表：sys.aud$中，也可从dba_audit_trail，dba_audit_statement，dba_audit_object中获得有关情况
  - 上述数据字典表需在DBA用户（system）下才可见

总结

- 事务是一个查询和更新的序列，他们共同执行某项任务。事务可以被提交或回滚。事务具有原子性、一致性、隔离性、持久性
- 完整性约束保证授权用户对数据库所做的改变不会导致数据一致性的破坏
  - 域完整性
  - 实体完整性（主键的约束）
  - 参照完整性（外键的约束）
  - 用户定义的完整性约束
- 断言是描述性表达式，它指定了我们要求总是为真的谓词
- 触发器定义了当某个事件发生而且满足相应条件时自动执行的动作

总结

- SQL的授权机制
- 角色有助于根据用户在组织机构中所扮演的角色，把一组权限分配给用户
- 审计跟踪用于跟踪安全漏洞或错误更新，oracle中的审计跟踪包括：
  - 语句审计
  - 对象（实体）审计

谢谢！