1. 什么是数据库的安全性？

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

2. 试述信息安全标准的发展史，试述 CC 评估保证级划分的基本内容。

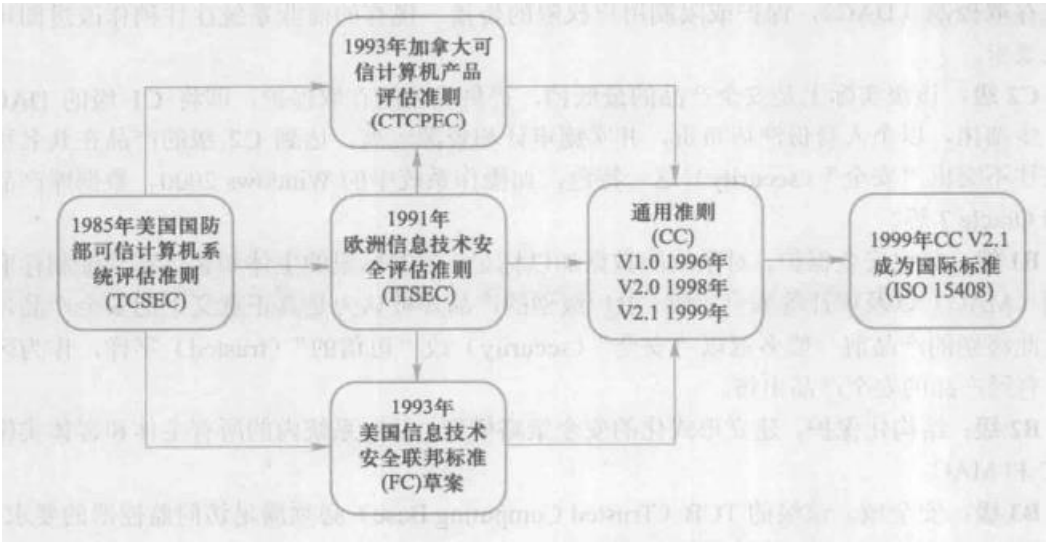


表 4.2　CC 评估保证级（EAL）的划分

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

3. TCSEC（TDI）安全级别等级划分的基本内容是什么？

   从四个方面来描述安全性级别划分的指标，即安全策略、责任、保证和文档。根据计算机系统对各项指标的支持情况，TCSEC/TDI 将系统划分为 4 组，7 个等级，依次是 D、C（C1，C2）、B（B1，B2，B3）、A（A1），按系统可靠或可信程度逐级增高。

表 4.1 TCSEC/TDI 安全级别划分

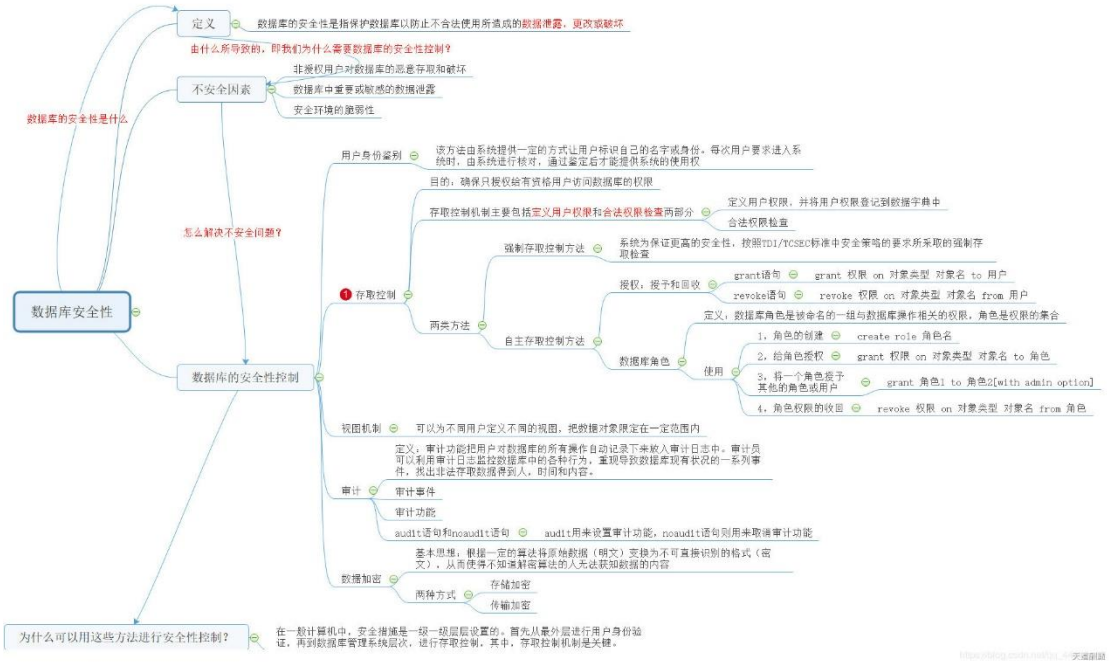
安全级别	定义
A1	验证设计（verified design）
B3	安全域（security domains）
B2	结构化保护（structural protection）
B1	标记安全保护（labeled security protection）
C2	受控的存取保护（controlled access protection）
C1	自主安全保护（discretionary security protection）
D	最小保护（minimal protection）

4. 数据库安全的技术手段有哪三类？试举例说明。(①预防保护类、②检测跟踪类和③响应恢复类)

预防保护（系统首先根据输入的用户标识进行用户身份鉴定，只有合法的用户才准许进入计算机系统）（用户身份鉴别）

检测跟踪（对已进入系统的用户，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，最后数据还能用加密的方式存储在数据库中，通过视图机制，为不同用户定义不同视图，把数据对象限定在一定的范围内）（多层存取控制、数据加密、视图）

响应恢复（根据具体安全需求配置审计规则，对用户访问行为和系统关键操作进行审计。通过设置简单入侵检测规则，对异常用户进行检测和处理）（审计，入侵检测）



5. SQL Server 中，常见的数据库安全策略包括哪些？

   1. 对 sql server2008 实例的访问权——登录名
   2. 对 sql server2008 数据库的访问权——数据库用户
   3. 对表和列的访问权——权限

6. 什么叫存储过程？

   存储过程是存储在服务器上的一组预编译的 SQL 语句。它可以包括数据操纵语句、变量、逻辑控制语句等。

7. 存储过程的优点

   （1） 执行速度快

   （2） 允许模块化设计

   （3） 提高系统安全性

   （4） 减少网络流量

8. 存储过程常见的类别（带参数，加密、组）

9. 存储过程对数据库的安全性有哪些帮助?

   可将存储过程作为用户存取数据的管道。建立特定的存储过程供用户使用完成对数据的访问。可以限制用户对数据表的存取权限。存储过程的定义文本可以被加密，使用户不能查看其内容

10. 如何用 SQL 语言进行存储过程创建和管理?

    见问题 8

11. 熟练进行存储过程的编写、调试。

    见问题 8

12. 熟练掌握应用 SQL Server 常见的系统存储过程。

    （1）*sp_addlogin：创建登录名。

    （2）*sp_droplogin：删除登录名。

    （3）sp_addrole：创建角色。

（4）*sp_adduser：创建用户。

（5）*sp_addsrvrolemember：将登录名添加到固定服务器角色。

（6）*sp_helplogins：查看每个数据库中的登录及相关用户的信息

13. 什么叫视图？视图对数据库安全的作用？

视图可以看作定义在 SQL Server 上的虚拟表。视图正如其名字的含义一样，是另一种查看数据的入口。

常规视图本身并不存储实际的数据，而仅仅是由 SELECT 语句组成的查询定义的虚拟表 。

为不同的用户定义不同的视图，把数据对象限制在一定的范围内。

14. 如何用 SQL 语言进行视图建立和管理？

```
CREATE [ OR ALTER ] VIEW [ schema_name . ] view_name [ (column [ ,...n ] ) ]
[ WITH <view_attribute> [ ,...n ] ]
AS select_statement
[ WITH CHECK OPTION ]
[ ; ]

<view_attribute> ::=
{
    [ ENCRYPTION ]
    [ SCHEMABINDING ]
    [ VIEW_METADATA ]
}
<select_statement> ::=
    [ WITH <common_table_expression> [ ,...n ] ]
    SELECT <select_criteria>
```

15. 触发器的概念，触发器与存储过程的关系？

（1） 触发器与存储过程非常类似：也是 SQL 语句集。两者的唯一的区别是触发器不能用 EXECUTE 语句调用，二十用户执行 T-SQL 语句时自动触发执行

（2） 触发器是一个在修改指定表中的数据时而执行的存储过程

（3） 触发器是不同于存储过程：触发器主要是通过事件进行触发而被执行的，而存储过程是可以通过存储过程的名字而被直接调用

16. 触发器的优点

（1） 触发器自动执行

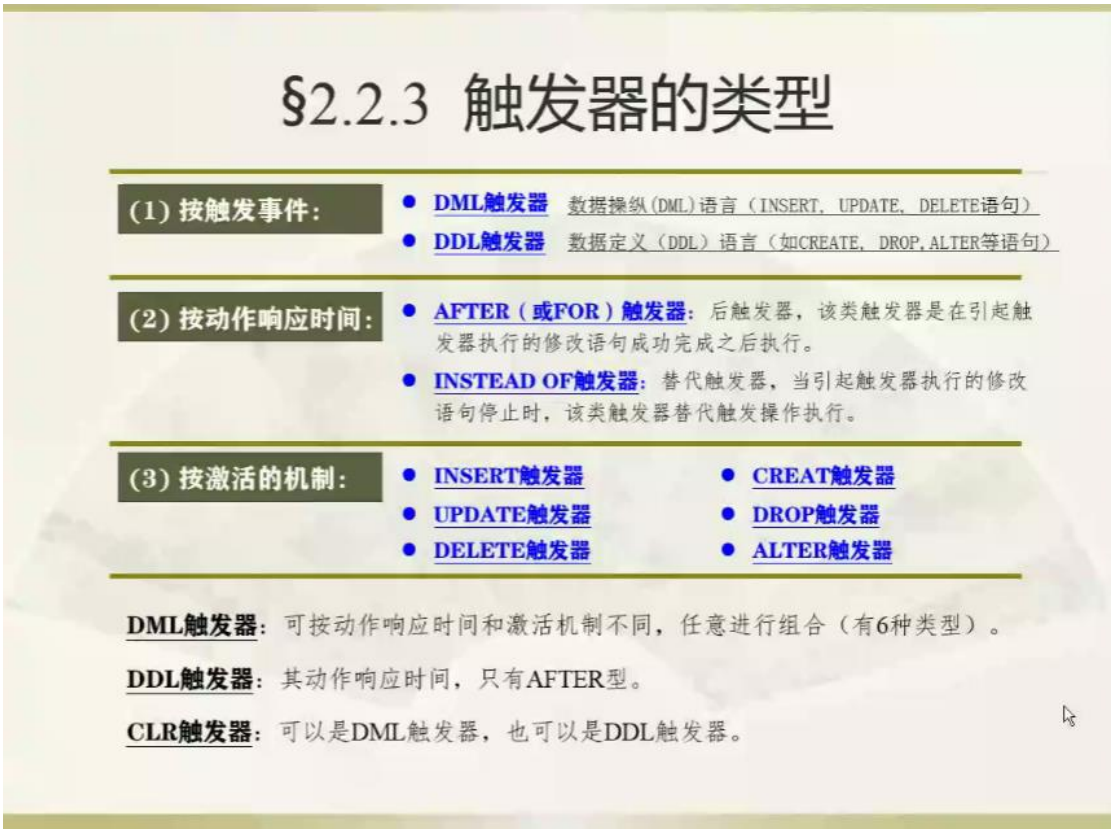
（2） 触发器能够对数据库中的相关表，实现级联更改

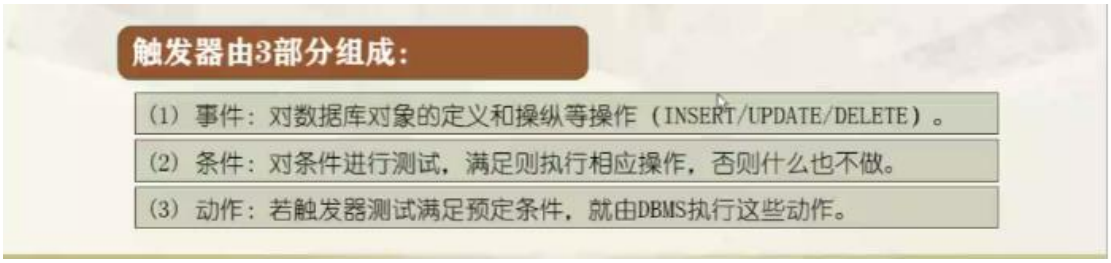
（3） 触发器可以实现比 CHECK 约束更为复杂的数据完整性约束

（4） 触发器可以评估数据修改前后的表的状态，并根据其差异采取对策

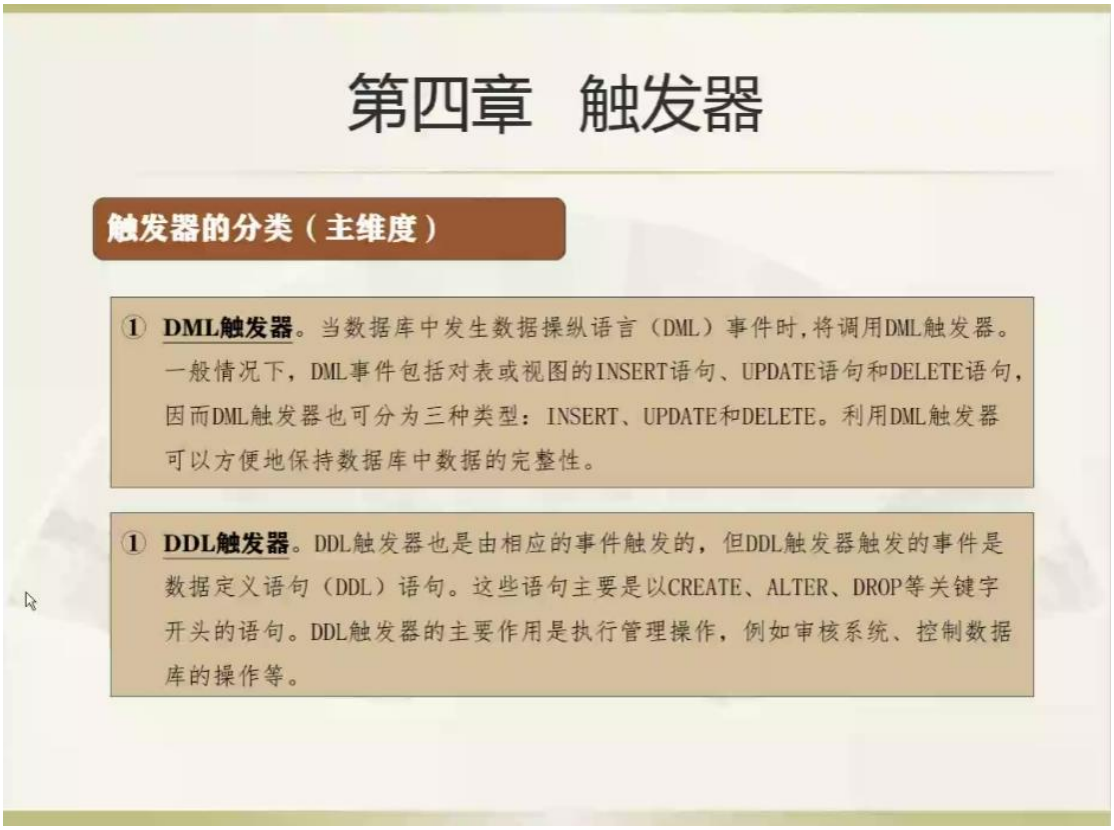
（5） 一个表中，可以同时存在三个不同操作的触发器。对于同一个修改语句，可以有多个不同的响应对策

17. 触发器的类型，触发器的组成。





18. DML 触发器，DDL 触发器。



19. AFTER 触发器，INSTEADOF 触发器


§2.2.4 DML触发器
触发器的动作相应时间：
(1) AFTER触发器(后触发器)：在触发器事件语句（INSERT、UPDATE、DELETE）执行成功之后，才执行触发器的动作语句。此类触发器不能创建在视图上。
可为每个触发器操作（INSERT、UPDATE、DELETE）创建多个AFTER触发器，使用系统存储过程sp_SetTriggerOrder设置哪个AFTER触发器最先激发（First触发器），哪个最后激发（Last触发器）。除第一个和最后一个触发器外，所有其他的AFTER触发器的激发顺序不确定，并且无法控制。
(2) INSTEAD OF触发器(替代触发器)：是在触发器事件语句（INSERT、UPDATE、DELETE）执行前，取消触发器事件语句，用触发器的动作语句来替代要执行的触发器事件语句完成操作。触发器表示并不执行其所定义的操作INSERT，UPDATE，DELETE，而仅是执行触发器本身。
该类触发器既可在表也可在视图上定义。每个触发操作（INSERT、UPDATE、DELETE），只能定义一个INSTEAD OF触发器。

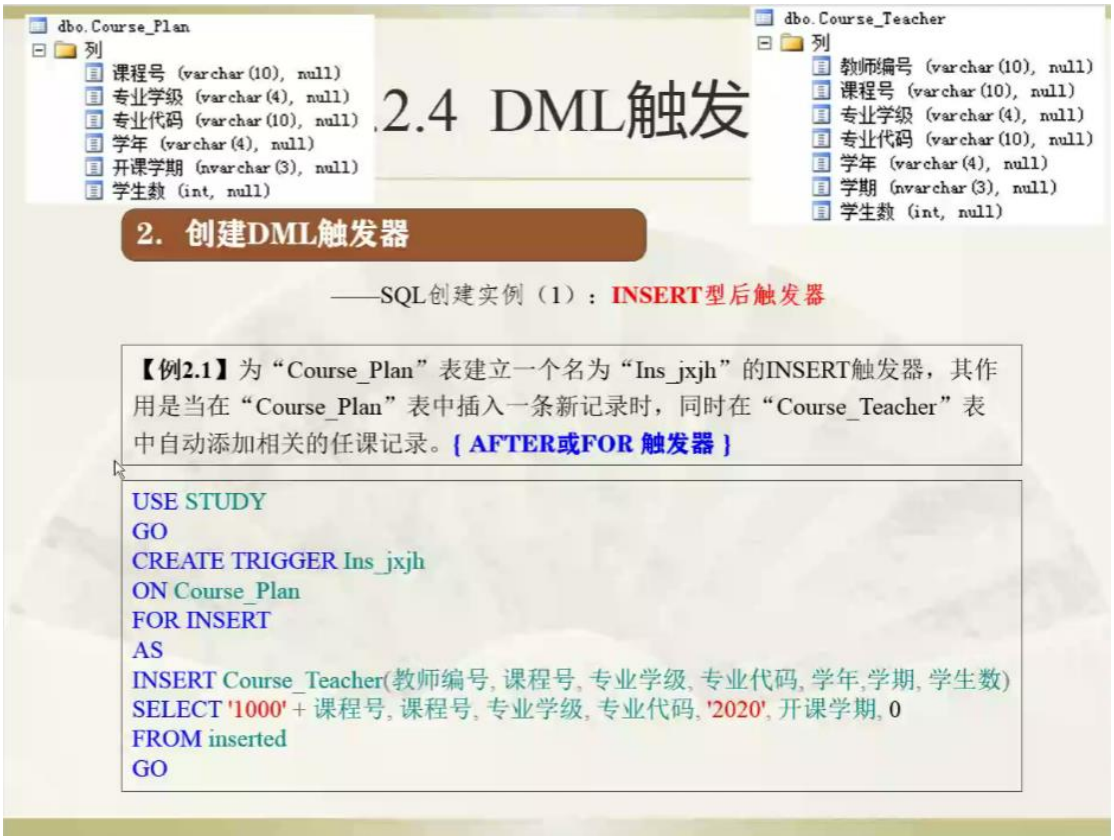
20. I 表\D 表


第二章 触发器
理解触发器里面的两个临时表：Deleted、Inserted。
注意：Deleted与Inserted分别表示触发事件的表“旧的一条记录”和“新的一条记录”。
一个数据库系统中有两个虚拟表，用于存储在表中记录改动的信息，分别是：
虚拟表Inserted
虚拟表Deleted
新增时：INSERT
存放新增的记录
不存储记录
修改时：UPDATE
存放用来更新的新记录
存放更新前的旧记录
删除时：DELETE
不存储记录
存放被删除的旧记录
一个Update的过程可以看作为：
①生成新的记录到Inserted表；
②复制旧的记录到Deleted表；
③接着删除Student表中的旧记录；
④最后向Student写入新纪录。
从前面的两个例子，
可以看到触发器的关键：
①2个临时的表；
②触发机制。

21. 修改记录=插入和删除

见问题 20

22. 触发器的创建、查看、修改、禁止和启用。



§2.2.4 DML触发器

2. 创建DML触发器

——SQL创建实例（2）：INSERT型替代触发器

【例2.1】的另外一种实现方法。{ INSTEAD OF 触发器 }

```
USE STUDY
GO
CREATE TRIGGER ins_jxjh_02
ON Course_Plan
INSTEAD OF INSERT
AS
BEGIN
    INSERT Course_Plan
      SELECT * FROM inserted
    INSERT Course_Teacher(教师编号, 课程号, 专业学级, 专业代码, 学年, 学期, 学生数)
      SELECT '1000' + 课程号, 课程号, 专业学级, 专业代码, '2020', 开课学期, 0
      FROM inserted
END
GO
```

§2.2.4 DML触发器

3. 查看触发器信息

——查看方法

(1) 工具（对象资源管理器）法：SQL Server Management Studio

右键选中“数据库”➔展开相应数据库的表节点➔选择某表➔下拉找“触发器”➔展开相应的触发器→右击选“修改”

(2) T-SQL（系统存储过程）法：

- sp_helptext 触发器名——查看触发器的文本信息
- sp_depends 触发器名——查看触发器的相关性
- sp_help 触发器名——查看触发器的一般信息
- sp_helptrigger 表名——查看表的触发器信息

23. 理解概念：①候选码、②主码、③外码、④主属性、⑤非主属性、⑥全码

若关系中的某一属性组的值能唯一地标识一个元组，而其子集不能，则称该属性组为**候选码**（candidate key）。

若一个关系有多个候选码，则选定其中一个为**主码**（primary key）。

候选码的诸属性称为**主属性**（prime attribute）。不包含在任何候选码中的属性称为**非主属性**（non-prime attribute）或**非码属性**（non-key attribute）。

在最简单的情况下，候选码只包含一个属性。在最极端的情况下，关系模式的所有属性是这个关系模式的候选码，称为**全码**（all-key）。

定义 2.5 设 F 是基本关系 R 的一个或一组属性，但不是关系 R 的码，K_s 是基本关系 S 的主码。如果 F 与 K_s 相对应，则称 F 是 R 的外码（foreign key），并称基本关系 R 为参照关系（referencing relation），基本关系 S 为被参照关系（referenced relation）或目标关系（target relation）。关系 R 和 S 不一定是不同的关系。

24. 数据库的完整性有：实体完整性（非空）、域完整性、参照完整性（外码，要么存在，要么空值）。

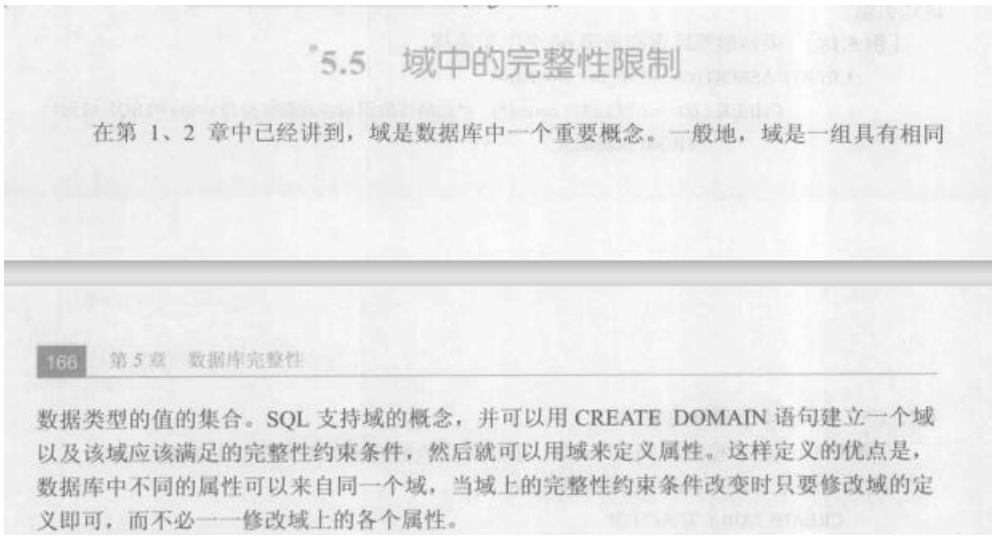
规则 2.1 实体完整性规则 若属性（指一个或一组属性）A 是基本关系 R 的主属性，则 A 不能取空值（null value）。所谓空值就是“不知道”或“不存在”或“无意义”的值。

规则 2.2 参照完整性规则 若属性（或属性组）F 是基本关系 R 的外码，它与基本关系 S 的主码 K_s 相对应（基本关系 R 和 S 不一定是不同的关系），则对于 R 中每个元组在 F 上的值必须：

- 或者取空值（F 的每个属性值均为空值）；
- 或者等于 S 中某个元组的主码值。

例如，对于例 2.1，学生关系中每个元组的“专业号”属性只能取下面两类值：

- 空值，表示尚未给该学生分配专业；
- 非空值，这时该值必须是专业关系中某个元组的“专业号”值，表示该学生不可能分配到一个不存在的专业中。即被参照关系“专业”中一定存在一个元组，它的主码值等于该参照关系“学生”中的外码值。



5.5 域中的完整性限制

在第 1、2 章中已经讲到，域是数据库中一个重要概念。一般地，域是一组具有相同

数据类型的值的集合。SQL 支持域的概念，并可以用 CREATE DOMAIN 语句建立一个域以及该域应该满足的完整性约束条件，然后就可以用域来定义属性。这样定义的优点是，数据库中不同的属性可以来自同一个域，当域上的完整性约束条件改变时只要修改域的定义即可，而不必一一修改域上的各个属性。

25. 数据库约束机制有哪些？（主键约束、外键约束、唯一性约束、检查约束、默认约束、非空约束）

§3.4 约束

§ 3.4.1 主键约束

1. 主键（码）

- 表的一列或几列的组合的值在表中唯一地指定一行记录，这样的一列或多列称为表的**主键（码）**（Primary Key，PK），通过它可强制表的实体完整性。

- **主键（码）**不允许为空值，且不同两行的键（码）值不能相同。表中可以有不止一个键（码）唯一标识行，每个键都称为**侯选键（码）**，只可以选一个侯选键作为表的主键（码）。

- 如果一个表的**主键由单列组成**，则该主键约束可以定义为该列的**列约束**。如果**主键由两个以上的列组成**，则该主键约束必须定义为**表约束**。

§3.4 约束

§ 3.4.2 外键约束

外键约束定义了表与表之间的关系。通过将一个表中一列或多列添加到另一个表中，创建两个表之间的连接，这个列就成为第二个表的外键(Foreign Key，FK)

外键是用于建立和加强两个表数据之间的连接的一列或多列，通过它可以强制参照完整性。

- 通过企业管理器在两个表之间建立外键关系
- 要提供两种级联操作，以保证数据完整性：
  - ① **级联删除**：确定当主键表中某行被删除时，外键表中所有相关行将被删除。
  - ② **级联修改**：确定当主键表中某行的键值被修改时，外键表中所有相关行的该外键值也将被自动修改为新值。

建立了外键约束后，被引用的表不能被删除

通过级联操作，保证引用表和被引用表之间数据的一致性和完整性

可以删除两张表之间的外键约束，然后可以删除被引用的表

§3.4 约束

§ 3.4.3 唯一性约束

唯一性(Unique)约束：指定一个或多个列的组合的值具有唯一性，以防止在列中输入重复的值，为表中的一列或者多列提供实体完整性。

唯一性约束指定的列可以有NULL属性。主键也强制执行唯一性，但主键不允许空值，故主键约束强度大于唯一约束。因此主键列不能再设定唯一性约束。

唯一性(Unique)约束用于非主键的列或者列组合

一张表只能定义一个主键约束，但是可以定义多个唯一性约束

§3.4 约束

§ 3.4.4 检查约束

检查(Check)约束对输入列或整个表中的值设置检查条件，以限制输入值，保证数据库的数据完整性。

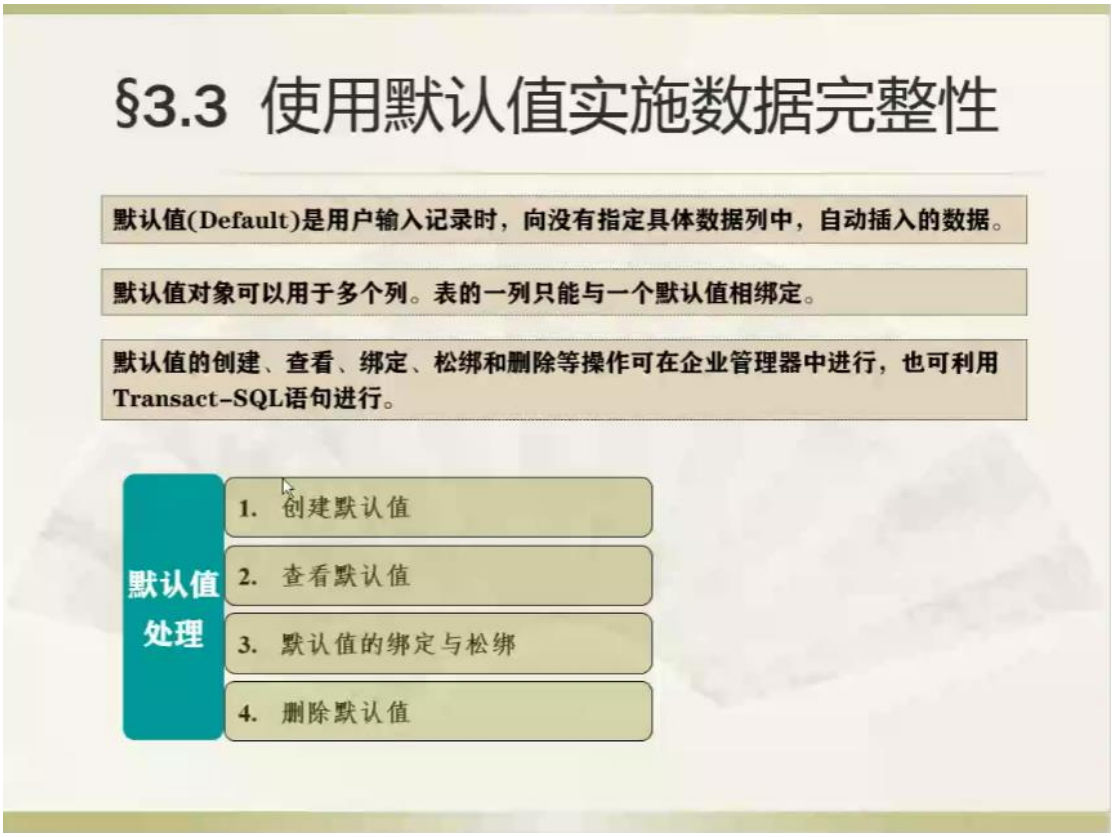
当对具有检查约束列进行插入或修改时，SQL Server将用该检查约束的逻辑表达式对新值进行检查，只有满足条件(逻辑表达式返回TRUE)的值才能填入该列，否则报错。

可以为每列指定多个CHECK约束。

检查约束的逻辑表达式可以使用表中的多列，但其必须定义为表级的检查约束

例如

```
Datediff(year, birth_date, hire_date)>18
```



非空约束，取值不能为 NULL

26. DBMS 的完整性控制机制应具有哪些功能？RDBMS 在实现参照完整性时需要考虑哪些方面？

（1） 提供定义完整性约束条件的机制

（2） 提供完整性检查的方法

（3） 进行违约处理

RDBMS 在实现参照完整性时需要考虑以下几个方面：

1） 外码是否可以接受空值

2）删除被参照关系的元组时的考虑，这时系统可能采取的作法有三种：

（a）级联删除（CASCADES）； （b）受限删除（RESTRICTED）；

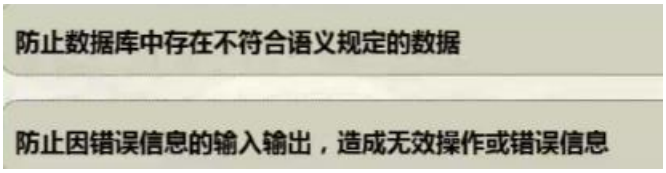
（c）置空值删除（NULLIFIES）

3） 在参照关系中插入元组时的问题，这时系统可能采取的作法有：

（a）受限插入 （b）递归插入

4） 修改关系中主码的问题 一般是不能用 UPDATE 语句修改关系主码的。如果需要修改主码值，只能先删除该元组，然后再把具有新主码值的元组插入到关系中。 如果允许修改主码，首先要保证主码的唯一性和非空，否则拒绝修改。然后要区分是参照关系还是被参照关系。

27. 约束/默认/规则的定义、查看、修改、绑定、松绑、删除。如何通过各种约束、默认、规则和触发器等数据库对象，来实施数据的完整性？约束/默认/规则并发实施完整性约束的逻辑？



28. 事务的概念及事务的四个特性？

事务是由有限数据库操作序列组成。为了保护数据库完整性，一般要求事务具有 4 个特征：原子性，一致性，隔离性，持久性

29. 多事务执行的方式有哪些？

事务串行执行，交叉并发方式，同时并发方式

30. 事务并发执行带来哪些问题？在数据库中为什么要并发控制？

可能会存取和存储不正确的数据，破坏事务的隔离性和数据库的一致性

DBMS 必须提供并发控制机制

并发控制机制，是衡量一个 DBMS 性能的重要标志之一

31. 并发控制的主要任务是什么？

对并发操作进行正确调度，保证事务的隔离性，保证数据库的一致性

32. 并发操作带来的数据不一致性包括哪三类？

丢失修改，不可重复读，读“脏”数据

33. 什么是封锁？基本的封锁类型有几种？试述它们的含义。锁的相容矩阵？

封锁就是事务 T 在对某个数据对象操作之前，先向系统发出请求，对其加锁

加锁后，事务 T 就对该数据对象有了一定的控制，在事务 T 释放它的锁之前，其它的事务不能更新此数据对象

封锁是实现并发控制的一个非常重要的技术

基本类型：排它锁，共享锁

排它锁：若事务 T 对数据对象 A 加上 X 锁，则只允许 T 读取和修改 A，其它任何事务都不能再对 A 加任何类型的锁，直到 T 释放 A 上的锁。

共享锁：若事务 T 对数据对象 A 加上 S 锁，则其它事务只能再对 A 加 S 锁，而不能加 X 锁，直到 T 释放 A 上的 S 锁。这就保证了其它事务可以读 A，但再 T 释放 A 上的 S 锁之前不能对 A 做任何修改。

T_2 \ T_1	X	S	-
X	N	N	Y
S	N	Y	Y
-	Y	Y	Y

34. 什么是封锁协议？三级封锁协议的概念与作用是什么？三级封锁协议的主要区别是什么？三级封锁协议各解决了什么问题？

在运用 X 锁和 S 锁对数据对象加锁时，需要约定一-些规定

则:封锁协议(Locking Protocol)何时申请 X 锁或 S 锁、持锁时间、何时释放

常用的封锁协议:三级封锁协议

不同的封锁协议，在不同的程度上为并发操作的正确调度提供一定的保证

1 级封锁协议：事务 T 在修改数据 R 之前，必须先对其加 X 锁，直到事务结束才释放（防止丢失修改）

2 级封锁协议= 1 级封锁协议+事务 T 再读取数据 R 前必须先加 S 锁，读完后即可释放 S 锁（防止丢失修改，读脏数据）

3 级封锁协议 = 1 级封锁协议+事务 T 在读取数据 R 之前必须先对其加 S 锁，直到事务结束才释放（防止丢失修改，读脏数据，不可重复读）

35. 什么是活锁与死锁？

活锁指的是任务或者执行者没有被阻塞，由于某些条件没有满足，导致一直重复尝试，失败，尝试，失败。

死锁：是指两个或两个以上的进程（或线程）在执行过程中，因争夺资源而造成的一种互相等待的现象，若无外力作用，它们都将无法推进下去。此时称系统处于死锁状态或系统产生了死锁，这些永远在互相等待的进程称为死锁进程。

36. 如何避免活锁？何谓一次封锁法/顺序封锁法？

采用先来先服务的策略

1)一次封锁法：就是要求每个事务必须一次将所有要使用的数据全部加锁，否则就不能继续执行，降低并发度。

2)顺序封锁法，预先对数据对象规定一个封锁顺序，所有事务都按这个顺序实行封锁。维护成本高，难以实现。

37. 如何死锁的预防？死锁的诊断与解除？

破坏死锁发生的条件。一次封锁法和顺序封锁法。允许死锁发生，解除死锁。检测方法：超时法，等待图法。解除死锁的方法：选择一个处理死锁代价最小的事务将其撤销，释放此事务持有的所有的锁，使其它事务能继续运行下去。对撤销事务所执行的数据修改操作，必须加以恢复

38. 什么样的并发操作调度是正确的？如何保证并发操作的调度是正确的？

几个事务的并行执行是正确的，当且仅当其结果与按某一次串行地执行它们时的结果相同。可串行性是并行事务正确性的唯一准则。方法：封锁方法（两段锁协议），时标方法，乐观方法。

39. 何谓两段封锁协议？

在对任何事务进行读、写操作之前，事务首先要获得对该数据的封锁。

在释放一个封锁之后，事务不再获得任何其它封锁。

40. 两段封锁协议与防止死锁的一次封锁法的关系？

一次封锁法遵守两段锁协议，遵守两段封锁协议的事务可能发生死锁。

41. 为什么说所有遵守两段锁协议的事务，其并行执行的结果一定是正确的？

因为遵守两段锁协议的事务所有的并行调度策略都是可串行化的

42. 两段封锁协议与三级封锁协议的关系？并发调度的正确性和数据的一致性有什么关系？为什么说遵守第三级封锁协议必然遵守两段协议？

两段封锁协议目的：保证并发调度的正确性。三级封锁协议目的：在不同程度上保证数据一致性。遵守第三级封锁协议必然遵守两段协议。并发调度的正确性保证数据的一致性。

43. 什么是封锁粒度？封锁粒度与被封锁的对象数、并发度、系统开销之间是什么关系？选择封锁粒度的原则是什么？

封锁对象的大小被称为封锁的粒度。

44. 什么是多粒度封锁？什么是三级粒度树？什么是多粒度封锁协议？

多粒度封锁：在一个系统中同时支持多种封锁粒度供不同的事务选择。

多粒度封锁规则：允许多粒度树中每个结点，被独立地加锁。对于一个

结点加锁意味着这个结点的所有后裔结点也被加以同样类型的锁。在多粒度封锁中，一个数据对象可能以两种方式封锁：显式封锁和隐式封锁。

45. 对某个数据对象加锁时，系统如何进行冲突检查/检查的内容有哪些？

该数据对象有无显示封锁与之冲突。所有上级结点：检查本事务的显示封锁，是否与该数据对象上的隐式封锁冲突。所有下级结点：看上面的显示封锁，是否与本事务的隐式封锁冲突。

46. 为什么要引进意向锁？意向锁的含义/作用是什么？

目的：提高对某个数据对象加锁时系统的检查效率。对任一结点加基本锁，必须先对它的上层结点加意向锁。如果对一个结点加意向锁，则说明该结点的下层结点正在被加锁。

47. 试述常用的意向锁：IS 锁、IX 锁、SIX 锁，给出这些锁的相容矩阵。

48. 数据库运行中可能产生的故障有哪几类？各种故障的定义？

事务内部故障（通过事务程序本身发现。有的是非预期的，不能由事务程序处理），系统故障（造成系统停止运转的任何事件，使得系统重新启动），介质故障（硬件故障），计算机病毒

49. 什么叫数据备份/转储？转储方式及转储状态分别有哪些？

为防止系统出现操作失误或者系统故障导致数据丢失，而将全系统或部分数据，从应用主机复制到其它存储介质上的过程。

转储方式	转储状态	
	动态转储	静态转储
海量转储	动态海量转储	静态海量转储
增量转储	动态增量转储	静态海量转储

50. 日志文件的作用是什么？日志文件有哪两种格式？每种格式的日志文件需要登记的内容包括哪些？登记日志文件的原则是什么？

日志文件在数据库恢复中起着非常重要作用，可以用来进行事务故障恢复和系统故障恢复，并协助后备副本进行介质故障恢复。主要的两种格式：以记录为单位的日志文件，以数据块为单位的日志文件。

以记录为单位的日志文件需要登记：每个事务的开始标记和结束标记，每个事务所有更新操作。

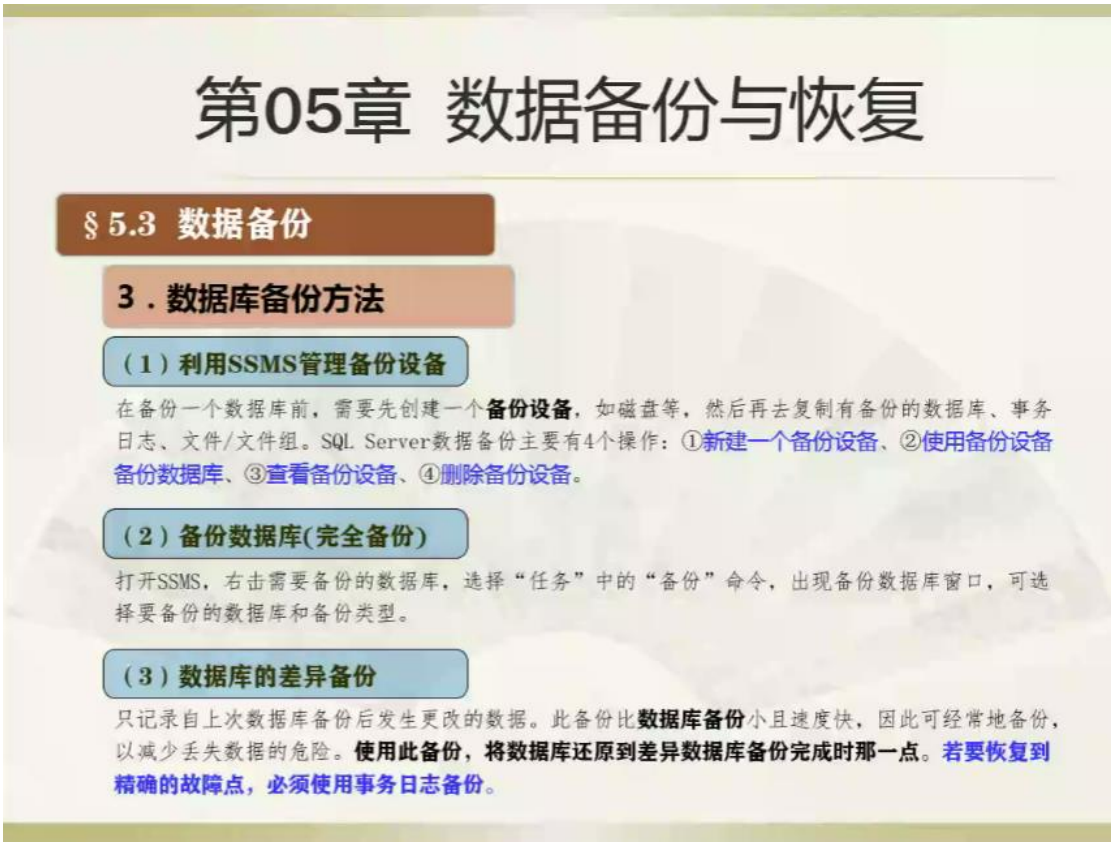
以数据块为单位的日志文件需要登记：事务标记，被更新的数据块。

原则：登记的次序严格按并发事务执行的时间次序，必须先写日志文件后写数据库

51. SQL Server 支持的备份类型有哪些？

完整备份，完整差异备份，部分备份，部分差异备份，文件和文件组备份，文件差异备份，事务日志备份。

52. 数据库备份方法（利用 SSMS 管理备份设备、备份数据库、数据库的差异备份）

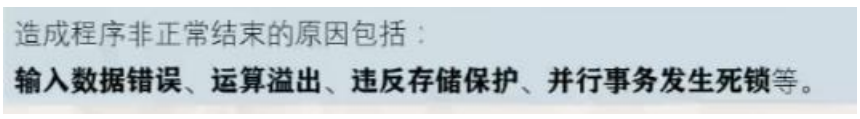


53. 使用 T-SQL 进行数据库备份

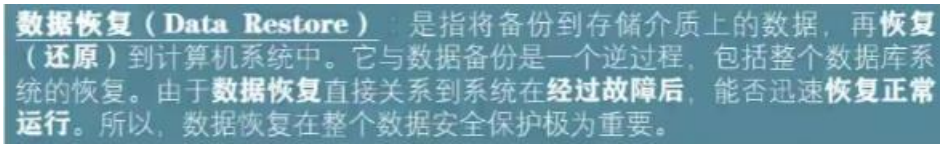
```
--完整备份
Backup   Database   db_database  To disk='D:\Backup\db_database_Full.bak'
--差异备份
Backup   Database   db_database  To
disk='D:\Backup\db_database_Diff.bak'  With  Differential
 解释如下:
      NAME = 'Full Backup of MyNwind'      --这个是备注，随便写。
 还原命令:
  USE master
  GO
  RESTORE DATABASE test_wt
      FROM disk = 'c:\test_wt'
  GO
```

54. 故障可分为 3 类？数据库故障产生的原因是什么？

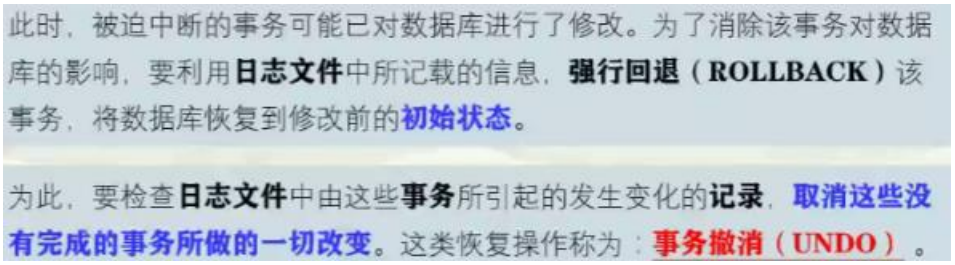
事务故障、系统故障、介质故障



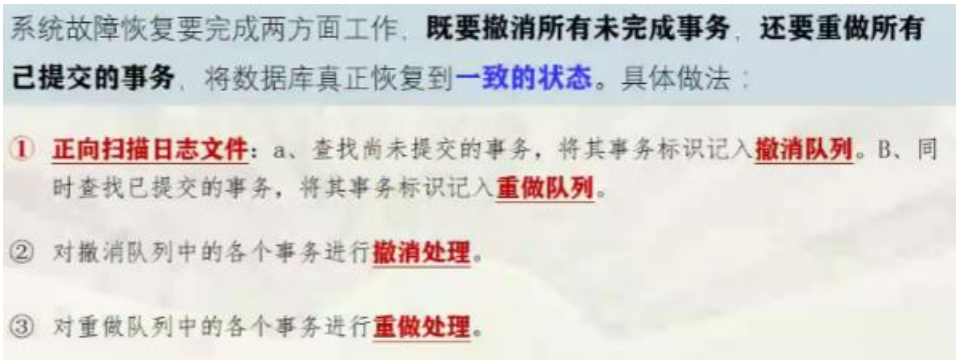
55. 什么叫数据恢复？针对不同故障（事务故障、系统故障、介质故障），有哪些恢复策略和方法。



事务恢复：

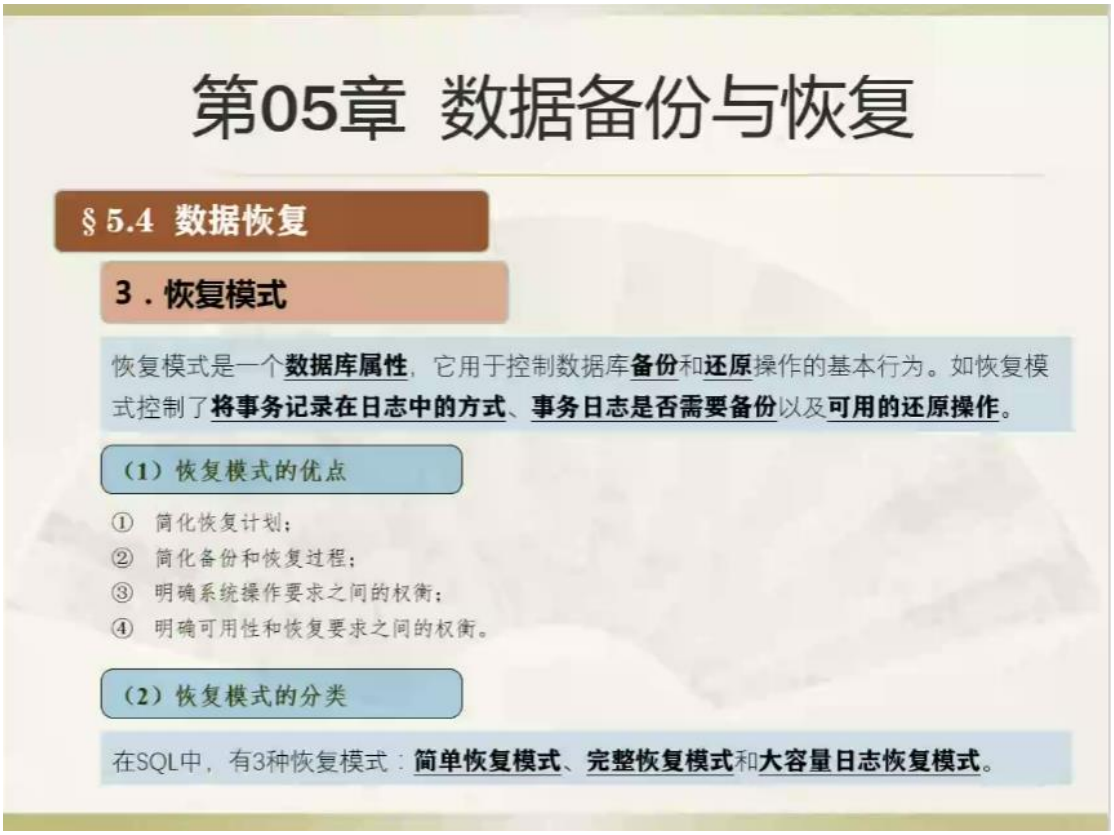


系统故障：

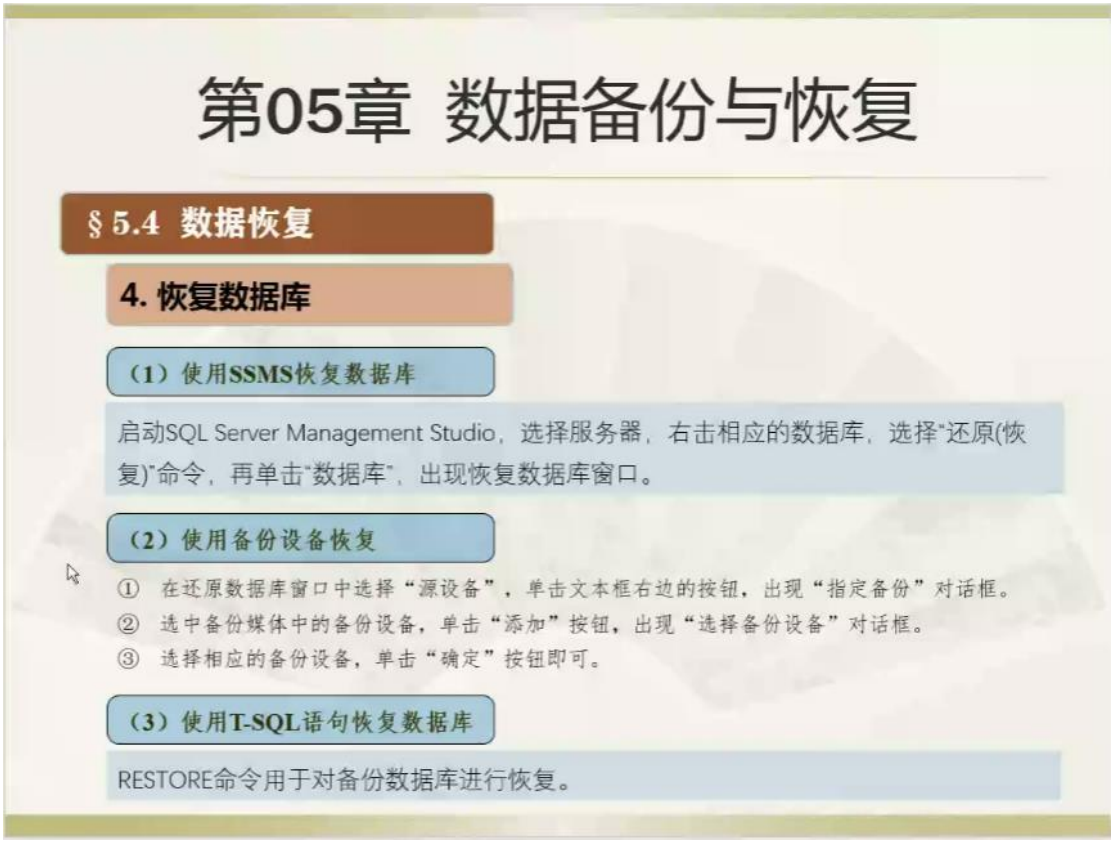


56. 数据恢复类型（全盘恢复、个别文件恢复、重定向恢复）

57. 数据恢复模式



58. 恢复数据库方法（使用 SSMS 恢复数据库、使用备份设备恢复数据库、使用 T-SQL 语句恢复数据库）



59. 使用 T-SQL 进行数据库恢复。

60. 什么是数据库镜像？数据库镜像有何用途？

数据库镜像即根据 DBA 的要求，自动把整个数据库或者其中的部分关键数据复制到另一个磁盘上。每当主数据库更新时，DBMS 自动把更新后的数据复制过去，即 DBMS 自动保证镜像数据与主数据的一致性.

用于数据库的恢复。当出现介质故障时候，DBMS 利用镜像磁盘数据进行数据库的恢复，同时镜像磁盘继续提供使用权，无需关闭系统和重装数据库副本。

提高数据库的可用性，用于并发操作。即当一个用户对数据加排它锁（写锁）修改数据时，其他用户可以读取镜像数据的数据，而不必等待该用户释放锁。

61. 概念：角色、用户、登录用户、数据库用户。

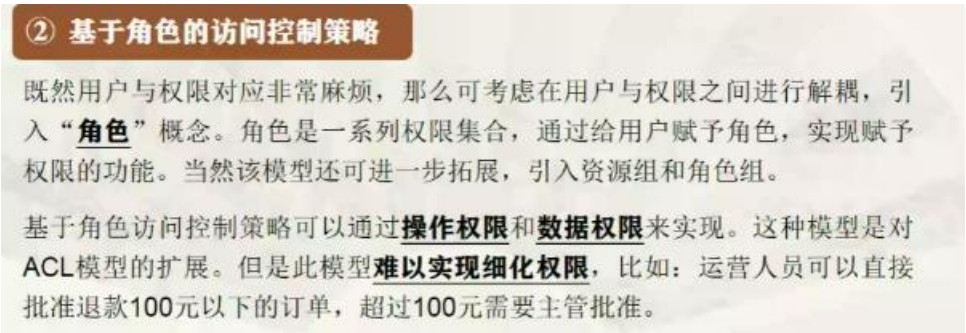
登录用户，指有权限登录到某服务器的用户，可以在有权限的情况下创建新的登录名，超级管理员的登录名是 sa；

服务器角色，指一组固定的服务器用户，默认有 9 组；

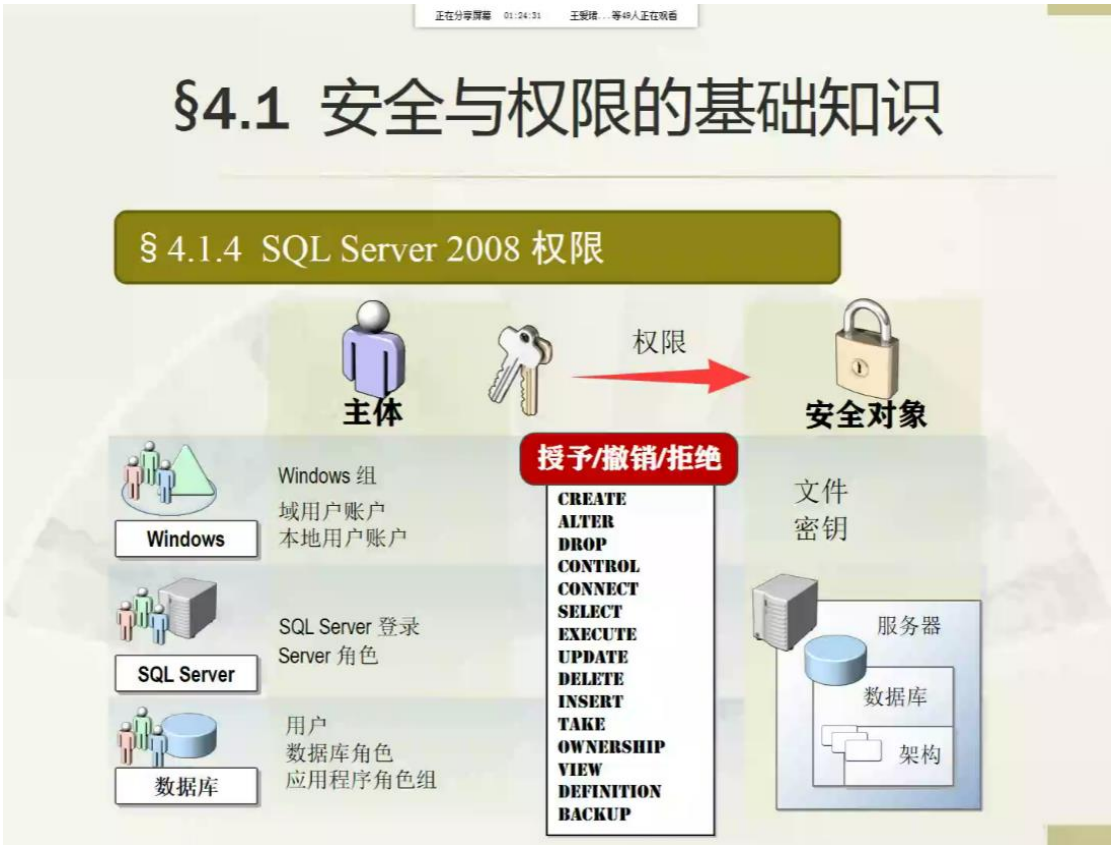
数据库用户，指有权限能操作数据库的用户；

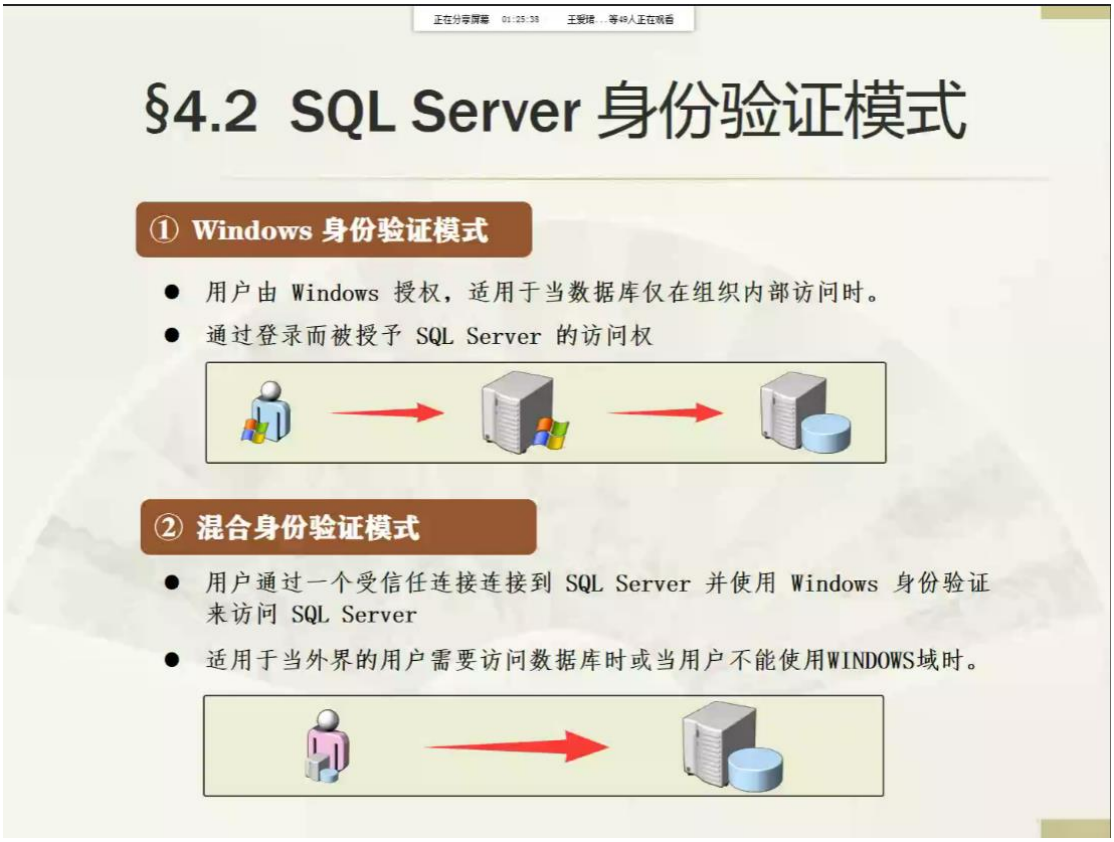
数据库角色，指一组固定的有某些权限的数据库角色；

62. 基于角色的访问控制如何实现权限控制？



63. SQL Server 访问控制与 SQL Server 身份验证模式是怎样的？





64. SQL Server 中，如何创建、修改、禁止和删除登录账户？固定服务器

角色有哪些？如何自定义服务器角色？

无法整理

65. SQL Server 中，如何创建、修改、禁止和删除数据库用户？固定数据库角色有哪些？如何自定义数据库角色？

无法整理

66. SQL Server 中，数据库权限分为哪三类？（①系统权限（隐含权限）、②对象权限、③语句权限）

67. SQL Server 中权限管理的内容包括哪些（授予权限、收回权限、拒绝权限，继承权限）。

68. 存取控制（自主存取控制、强制存取控制）。

69. 熟练使用 GRANT 和 REVOKE 语句完成授权定义和存取控制功能。

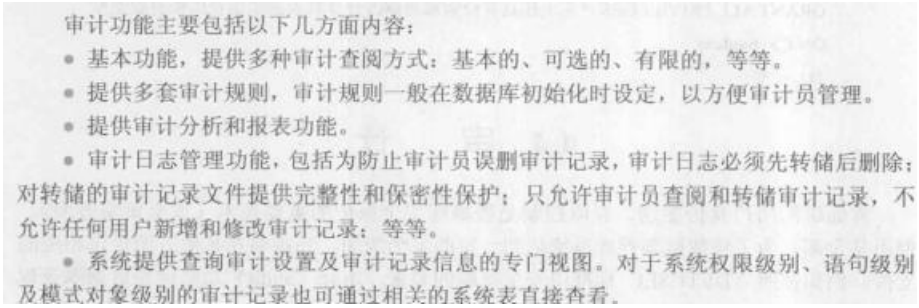
70. 什么是数据库审计功能？

审计功能把用户对数据库的所有操作自动记录下来放入审计日志。审计员可以利用审计日志监控数据库中的各种行为，重现导致数据库现有状况的一系列事件，找到非法存取数据的人、时间和内容等。

71. 为什么要提供审计功能？

使数据库管理系统达到一定的安全级别。提供一种事后检查的安全机制

72. 数据库的审计系统的主要功能。



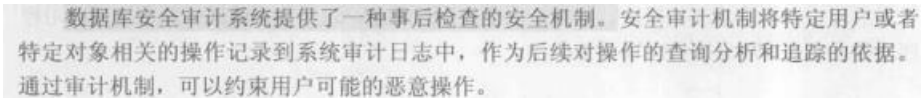
审计功能主要包括以下几方面内容：

- 基本功能，提供多种审计查阅方式：基本的、可选的、有限的，等等。
- 提供多套审计规则，审计规则一般在数据库初始化时设定，以方便审计员管理。
- 提供审计分析和报表功能。
- 审计日志管理功能，包括为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；等等。
- 系统提供查询审计设置及审计记录信息的专门视图。对于系统权限级别、语句级别及模式对象级别的审计记录也可通过相关的系统表直接查看。

73. 数据库审计系统的主要特点。

没找到。

74. 审计系统的建设目标是什么?


数据库安全审计系统提供了一种事后检查的安全机制。安全审计机制将特定用户或者特定对象相关的操作记录到系统审计日志中，作为后续对操作的查询分析和追踪的依据。通过审计机制，可以约束用户可能的恶意操作。

75. 什么是 SQL 注入攻击?

SQL 注入式攻击，就是攻击者把 SQL 命令插入到 Web 表单的输入域或页面请求的查询字符串，欺骗服务器执行恶意的 SQL 命令

76. SQL 注入攻击有何有何特点和危害?

特点：变种极多，攻击简单，危害极大

危害：未经授权状况下操作数据库中的数据，恶意篡改网页内容或者是数据库账号，私自添加系统账号或者是数据库使用者账号，网页挂木马。

77. 如何防御 SQL 注入攻击?

构造的 sql 语句时使用参数化形式而不使用拼接方式能够可靠地避免 sql 注入。