第5章

数据库安全性

5.1 数据库安全性概述


微课视频

安全性对于任何一个数据库系统来说都是至关重要的。通常，数据库中存储了大量的数据，这些数据可能是个人信息或其他保密资料。如果有人未经授权非法侵入了数据库，并窃取了查看和修改数据的权限，将会造成极大的伤害。

有时人们将数据库安全性问题与数据完整性问题混淆，但实际上这是两个不同的概念。安全性是指保护数据以防止不合法的使用而造成数据被泄露、更改和破坏；完整性是指数据的准确性和有效性。也就是说，安全性保护数据以防止不合法用户故意造成的破坏；完整性保护数据以防止合法用户无意中造成的破坏。安全性确保用户被允许做其想做的事情；完整性确保用户所做的事情是正确的。

5.1.1 安全标准

计算机及安全性技术方面有一系列的安全标准，最有影响的是 TCSEC 标准和 CC 标准。

1. TCSEC 标准

TCSEC 标准是指 1985 年美国国防部颁布的《可信计算机系统评估标准》(Trusted Computer System Evaluation Criteria，TCSEC)，又称橘皮书。TCSEC 从安全策略、责任、保证、文档四个方面来描述安全性级别划分的指标。根据计算机系统对各项指标的支持情况，TCSEC 将系统划分为四个等级，从低向高依次是 D、C1、C2、B1、B2、B3、A1。如表 5.1 所示，各安全级别之间，偏序向下兼容。

表 5.1 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)

对于 B2 级别以上的系统应用多限于一些特殊的部门。一些安全研究试图将目前仅限于少数领域应用的 B2 安全级别应用到商业中，并逐步成为新的商业标准。

2. CC 标准

CC 标准具有结构开放、表达方式通用等特点。CC 标准提出了国际公认的结构，以此结构来表述信息技术安全性，把信息产品的安全要求分为安全功能要求和安全保证要求。

CC 标准的文本由三部分组成，分别是：简介和一般模型、安全功能要求，以及安全保证要求。简介和一般模型，介绍了 CC 标准中的有关术语、基本概念和一般模型以及与评估有关的一些框架；安全性功能要求，列出了一系列类、子类和组件；安全保证要求，根据系统对安全保证要求的支持情况提出了评估保证级别。

CC 标准评估保证级别划分为 7 级，从 EAL1 至 EAL7，保证程度逐渐增高，如表 5.2 所示。

表 5.2 CC 评估保证级别划分

保证级别	定　义	近似相当 TCSEC 安全级别
EAL1	功能测试(functionally tested)	
EAL2	结构测试(structurally tested)	C1
EAL3	系统的测试和检查(methodically tested and checked)	C2
EAL4	系统的设计、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

5.1.2 数据库安全性的类型和一般措施

数据库的安全性是一个涵盖许多问题的广阔领域，主要包括以下几项。

1. 某些信息的访问关系到法律和伦理的问题

例如，有些信息可能会认为是属于私人信息，未授权人员不能合法地对其进行访问。

2. 有关政府、机构或企业公司层面的一些政策问题

例如，这些政策有时会确定哪些信息不应该向公众公开。

3. 与系统有关的问题

例如，在系统级上规定应该加强哪些安全功能。

4. 与安全性级别有关的问题

例如，一些组织需要把安全性问题划分为多个安全级别，并基于这些级别对数据和用户进行划分。

数据库的安全性威胁会使数据库的完整性、可用性、机密性等目标削弱或丧失。在一个多用户数据库系统中，DBMS 必须提供相应的技术以保证特定的用户或用户组只能访问数据库的指定部分，而不能访问数据库的其他部分。

一般地，一个典型的 DBMS 包含一个数据库安全和授权子系统，由它来负责实现一个数据库的安全性功能，从而避免发生未授权的访问。目前一般涉及自主安全机制和强制安全机制两种类型的数据安全机制。自主安全机制用于向用户授予特权，强制安全机制用于对多级安全性进行控制。

为了保证数据库的安全性，常用的控制措施主要有访问控制、推理控制和数据加密。

1）访问控制

DBMS 的安全机制必须包括限制对数据库系统整体访问的规定。这个功能称为访问控制，它的实现是由 DBMS 通过创建用户账号和口令的方式来控制登录过程。

2）推理控制

推理控制措施对应的安全性被称为统计数据库安全性，统计数据库是用于提供基于各种标准的统计信息和值的汇总数据，其安全性必须确保有关个人的信息不能被访问。有时可能仅仅从一些用户组有关的统计信息查询中就可以推导出涉及个人信息的数据。所以这种查询必须被禁止。

3）数据加密

数据加密可以用于为数据库的敏感部分提供额外的保护措施。例如，可以使用一些编码加密算法对数据进行编码，使未授权用户很难破译已经被编码的数据，已经授权的用户则可以通过解密算法来破译数据。

当数据库面对安全性问题时，数据库管理系统自身并不能完全担负起维护数据库安全性的任务，需要应用程序、网络服务、安全监控等多部分联合作用。

5.1.3　数据库安全的威胁

为了保证数据库安全，系统的所有部分都必须是安全的，因此全面的数据库安全计划必须考虑下列情况。

1. 损失可用性

损失可用性表示用户不能访问数据或系统，破坏硬件、网络或应用程序会导致数据库可用性损失，会使系统出现严重问题。

2. 损失机密性数据

损失机密性数据是指数据库中涉及企业或机构的一些关键性数据损失，这种损失会使企业失去竞争力。

3. 损失秘密性数据

损失秘密性数据是指数据库中涉及个人的一些数据损失，这种损失会导致对个人不合理的行为。

4. 偷窃和诈骗

偷窃和诈骗不仅影响数据库环境，而且影响企业的正常运营。一般来说，这种情况与人有关，所以需要集中精力减少这种活动的发生。

5. 意外损害

意外损害一般是非蓄意造成的，包括人为的错误、软硬件引起的破坏等。所以，数据库安全性的目标是保护数据免受意外或故意的丢失、破坏或滥用。


微课视频

5.2 数据库安全性控制

5.2.1 用户标识与鉴别

数据库管理员（DataBase Administrator，DBA）负责数据库系统的安全。因此，DBA 必须能够识别安全威胁，能够实施安全措施，从而使可能发生的数据库安全威胁最小化。

任何需要访问数据库系统的一个用户或一组用户都必须首先向 DBA 申请账户。然后 DBA 基于合理需求和相关约束规定，为用户创建访问数据库的账户和口令。以后，当用户需要访问数据库时，需要使用给定的账户和口令登录，DBMS 核对登录账号的有效性之后，允许用户访问数据库。

在一般的计算机系统中，安全措施是一级一级层层设置，如图 5.1 所示。

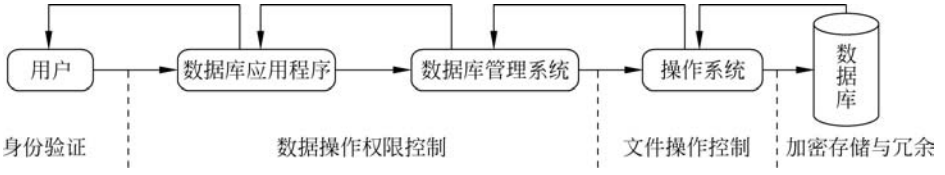

图 5.1 安全性控制模型

当用户要访问数据库数据时，应该首先进入数据库系统。用户进入数据库系统通常是通过数据库应用程序实现的，这时用户要向数据库应用程序提供其身份（如用户名和密码），然后数据库应用程序将用户的身份递交给数据库管理系统进行验证，只有合法的用户才能进入到下一步的操作。

对于合法的用户，当其要在数据库中执行某个操作时，数据库管理系统验证此用户是否具有执行该操作的权限。如果有操作权限，才执行操作，否则拒绝执行此用户的操作。

在操作系统一级也可以有自己的保护措施，如设置文件的访问权限等。

对于存储在磁盘上的数据库文件可以进行加密存储，保证数据如果被窃取也很难获取信息；同时还可以将数据库文件进行冗余备份，保证如果出现意外情况可以找到副本数据进行恢复。

5.2.2 存取控制策略

现在的 DBMS 通常采用自主存取控制和强制存取控制两种方法来解决数据库安全系统的访问控制问题。无论采用哪一种存取控制方法,需要保护的数据单元或数据对象包括从整个数据库到某个元组的某个部分。

存取控制主要包括定义用户权限和合法权限检查两部分。

1. 定义用户权限

用户对某一数据对象的操作权利称为权限。某个用户应该具有何种权限是管理和政策层面的问题并非技术问题。数据库管理系统的功能是保证这些决定的执行。因此,数据库管理系统需要提供适当的语言来定义用户权限。

2. 合法权限检查

每当用户发出存取数据库的操作请求后,数据库管理系统进行合法权限检查,如果用户提出的操作请求超出了定义的权限,系统将拒绝执行此操作。

定义用户权限和合法权限机制一起组成了数据库管理系统的存取控制子系统。C2 级的数据库管理系统支持自主存取控制,B1 级的数据库管理系统支持强制存取控制。

自主存取控制是指用户对不同的数据对象具有不同的存取权限,而且没有固定的关于哪些用户对哪些对象具有哪些存取权限的限制。所以自主存取控制非常灵活。

强制存取控制是指对每一个数据对象都被标识一定的密级,每一个用户也被授予一个许可证级别。对于任意一个数据对象,只有具有合法许可证级别的用户才可以存取。所以强制存取控制本质上具有分层的特点且相对比较严格。

不管采用自主存取控制还是强制存取控制方法,所有关于哪些用户可以对哪些数据对象进行操作的决定都是由政策和管理层面决定的,并非数据库管理系统决定,数据库管理系统只是实施这些决定。

5.2.3 自主存取控制

数据库系统中实施自主存取控制的典型方法是基于授权和收回权限的机制。授予和收回权限是数据库管理员(DBA)的职责。DBA 依据数据的实际应用情况将合适的权限授予给相应的用户。

1. 自主权限的类型

DBMS 必须能够基于指定的账户,提供对数据库中每个关系的选择性访问,能够控制用户执行的操作。因此,在使用数据库系统时可以在两个级别上分配权限。

(1) 用户级:DBA 可以为每个用户指定其独立持有的在数据库关系上的特定权限。

在用户级上的权限可以为该用户提供一些能力,包括用于创建模式或基本关系的权限,用于创建视图的权限,用于支持在关系中增加或删除属性等模式变化的权限,用于删除关系或视图的权限,用于插入、删除和修改元组的权限,用于从数据库中获取信息的权限等。

(2) 关系级:DBA 可以控制数据库中每个单独的关系或视图的访问权限。

关系级别上的权限用于对每一个用户指定在每个单独的关系上适用哪些类型的命令。有些权限还可以作用于关系中单独的属性。为了控制关系权限的授予和收回，数据库中的每个关系都被分配一个属主账户，属主账户一般是第一次创建该关系的用户使用的账户。关系的属主拥有这个关系上的所有权限。在 SQL 中，每个单独的关系可以被授予关系上的选择权限、修改权限和参照权限。

2. 权限的授予和收回

1）权限的授予

主要是通过 SQL 的 GRANT（授予权限）语句来完成权限的授予，另外，视图机制也是一种重要的自主性授权机制。

2）权限的收回

主要是通过 SQL 的 REVOKE（收回权限）语句来完成权限的收回。

3）使用 GRANT OPTION 选项传播权限

每当关系的拥有者 A 把关系上的一个权限授予另一个账户 B 的时候，可以给账户 B 授予带有 GRANT OPTION 选项和不带有 GRANT OPTION 选项的特权。如果带有这个选项，表示账户 B 还可以把关系上拥有的权限授予其他账户。

【例 5-1】 使用 GRANT 语句创建一个新的用户 testUser，密码为 testPwd。用户 testUser 对所有的数据有查询、插入权限，并授予 GRANT 传播权限。

使用 GRANT 语句授权，语句如下。

```
mysql > GRANT SELECT, INSERT ON * . *
    - > TO 'testUser'@'localhost'
    - > IDENTIFIED BY 'testPwd'
    - > WITH GRANT OPTION;
Query OK, 0 rows affected, 1 warning (0.05 sec)
```

使用 SELECT 语句查询用户 testUser 的权限，语句如下。

```
mysql > SELECT Host, User, Select_priv, Grant_priv
    - > FROM mysql.user
    - > WHERE User = 'testUser';
+-----------+----------+-------------+-------------+
| Host      | User     | Select_priv | Grant_priv  |
+-----------+----------+-------------+-------------+
| localhost | testUser | Y           | Y           |
+-----------+----------+-------------+-------------+
```

5.2.4 强制存取控制

自主存取控制能够通过授权机制来有效控制对敏感数据的存取，但由于用户对数据的存取是“自主”的，用户可以自由地决定将数据的存取权限或者将“授权”权限授予其他用户。在这种授权机制下，仍可能存在数据的“无意泄露”，造成这种“无意泄露”的根本原因在于这

种机制仅通过对数据的存取权限来进行安全控制，而数据本身并没有安全性标记。此时就需要对系统控制下的所有主客体实施强制存取控制策略。

在强制存取控制中，DBMS将全部实体划分为主体和客体两大类。主体是系统中的活动实体，包括DBMS所管理的实际用户以及代表用户的各个进程。客体是系统中的被动实体，是受主体操作的，包括文件、基本表、索引、视图等。对于主体和客体，DBMS为每个实例指派一个敏感度标记(Label)。

敏感度标记被分为若干级别，典型的安全性级别有：绝密(Top Secret，TS)、机密(Secret，S)、秘密(Confidential，C)和无分类(Unclassified，U)。其中，TS是最高级别，U是最低级别，级别从高到低依次是TS、S、C、U。

主体的敏感度标记被称为许可证级别(Clearance Level)，客体的敏感度标记被称为密级(Classification Level)。强制存取控制就是对比主体的Label和客体的Label，最终确定主体是否能够存取客体。

这种基于主体/客体安全性级别的多级安全性的强制存取控制策略有以下两个限定规则。

规则1：仅当主体的许可证级别大于或等于客体的密级时，该主体才能读相应的客体。

规则2：仅当主体的许可证级别等于客体的密级时，该主体才能写相应的客体。

在有些系统中，规则略有差别，被称为简单安全性和星性质。简单安全性是指安全性级别低的主体不能读取安全性级别高的客体。星性质是指禁止主体写安全性级别比它的安全性级别低的客体，如果违反了这个规则，那么就是允许信息从较高的安全性级别流向较低的安全性级别。

强制存取控制是对数据本身进行密级标记，无论数据如何被控制，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能操作数据，从而提高了安全性。

5.2.5　视图机制

视图是一个虚表，包含一系列带有名称的行和列数据。视图中数据来自定义视图时的查询所引用的表，并且在引用视图时动态生成。在安全性控制上，为不同的用户定义特定的数据和所负责的特定任务，建立不同的视图，把数据对象限制在一定的范围内，通过视图机制把要保密的数据对无权存取的用户隐藏起来。

1. 创建视图

在MySQL中可以使用CREATE VIEW语句来创建视图，其语法格式如下。

```
CREATE VIEW view_name
AS SELECT 语句
[WITH CHECK OPTION]
```

其中，view_name是指定的视图的名称，该名称在数据库中必须是唯一的，不能与其他表或视图同名。SELECT语句是用于指定创建视图的，这个SELECT语句给出了视图的定义，它可用于查询多个基础表或源视图。WITH CHECK OPTION为可选子句，用于指定在可更新视图上所进行的修改都需要符合SELECT语句中所制定的限制条件。

对于 SELECT 语句的指定，存在以下限制。

（1）定义视图的用户除了要求被授予 CREATE VIEW 的权限之外，还必须被授予可以操作视图所涉及的基础表或其他视图的相关权限，例如，由 SELECT 语句选择的每一列上的某些权限。

（2）SELECT 语句不能包含 FROM 子句中的子查询。

（3）SELECT 语句不能引用系统变量或用户变量。

（4）SELECT 语句不能引用预处理语句参数。

（5）在 SELECT 语句中引用的表或视图必须存在。

（6）若 SELECT 语句中所引用的不是当前数据库的基础表或源视图时，需要在该表或视图前加上数据库的名称作为限定前缀。

（7）在由 SELECT 语句构造的视图定义中，允许使用 ORDER BY 子句，但是如果从特定视图进行了选择，而该视图使用了自己的 ORDER BY 语句，则视图定义中的 ORDER BY 子句被忽略。

【例 5-2】 在学生-课程 -成绩数据库中创建视图 v_student，要求该视图中包含学生信息表 S 中所有男生的信息，并且要求保证以后对该视图数据的修改都必须符合“学生性别为男”这个条件。

完成此操作的 SQL 语句如下。

```
mysql > CREATE VIEW v_student
      - > AS
      - > SELECT * FROM S WHERE Ssex = '男'
      - > WITH CHECK OPTION
Query OK, 0 rows affected (015 sec)
```

【例 5-3】 在学生-课程-成绩数据库中创建视图 v_grade_avg，要求该视图中包含成绩表 SC 中所有学生的学号和平均成绩。

完成此操作的 SQL 语句如下。

```
mysql > CREATE VIEW v_grade_avg
      - > AS
      - > SELECT Sno, AVG(Grade) FROM SC
      - > GROUP BY Sno
Query OK, 0 rows affected (0.09 sec)
```

【例 5-4】 在学生-课程-成绩数据库中创建视图 v_grade，要求该视图中包含成绩表 SC 中所有学生成绩小于 90 的学生学号、课程号和成绩。

完成此操作的 SQL 语句如下。

```
mysql > CREATE VIEW v_grade
      - > AS
      - > SELECT * FROM SC WHERE Grade < 90
      - > WITH CHECK OPTION
Query OK, 0 rows affected (0.09 sec)
```

2. 删除视图

在 MySQL 中，可以使用 DROP VIEW 语句来删除视图，其语法格式如下。

```
DROP VIEW view_name [,view_name] …
```

其中，view_name 指定要被删除的视图名。使用 DROP VIEW 语句可以一次删除多个视图，但必须在每个视图上拥有 DROP 权限。

【例 5-5】 删除学生-课程-成绩数据库中的视图 v_student。

完成此操作的 SQL 语句如下。

```
mysql > DROP VIEW v_student;
Query OK, 0 rows affected (0.05 sec)
```

3. 修改视图定义

在 MySQL 中，可以使用 ALTER VIEW 语句来对已有视图的定义或结构进行修改，其语法格式如下。

```
ALTER VIEW view_name
AS SELECT 语句
[WITH CHECK OPTION]
```

ALTER VIEW 语句的语法同 CREATE VIEW 类似，因此不再重复描述。但需要注意的是，对于 ALTER VIEW 语句的使用，需要用户拥有视图的 CREATE VIEW 和 DROP 权限，以及由 SELECT 语句选择的每一列上的某些权限。

【例 5-6】 使用 ALTER VIEW 语句修改学生-课程-成绩数据库中的视图 v_student 的定义，要求该视图包含学生信息表 S 中“学生性别”为“男”，“年龄”小于“20 岁”的学生的学号、姓名和所属班级，并且要求今后对该视图的修改都必须符合“学生性别”为“男”，年龄”小于“20 岁”这个条件。

完成此操作的 SQL 语句如下。

```
mysql > ALTER VIEW v_student (Sno,Sname,Cno);
    - > AS
    - > SELECT Sno,Sname,Cno FROM S
    - > WHERE Ssex = '男' AND Sage < 20;
    - > WITH CHECK OPTION
Query OK, 0 rows affected (0.18 sec)
```

另外，修改视图的定义也可以通过先使用 DROP VIEW 语句，再使用 CREATE VIEW 语句的过程来实现。

4. 查看视图定义

在 MySQL 中可以使用 SHOW CREATE VIEW 语句来查看已有视图的定义或结构，其语法格式如下。

```
SHOW CREATE VIEW view_name
```

其中，view_name 指定要查看视图的名称。

5. 更新视图数据

由于视图是一个虚拟表，所以通过插入、修改和删除等操作方式来更新视图中的数据，实质是更新视图所应用的基础表中的数据。然而，视图的更新操作是受一定限制的，并非所有的视图都可以进行插入（INSERT）、更新（UPDATE）或删除（DELETE）等操作，只有满足可更新条件的视图才能进行更新，否则就会导致系统出现不可预期的结果。

对于可更新的视图，要求该视图中的行和基础表中的行之间具有一对一的对应关系。另外，如果视图中包含聚合函数、DISTINT 关键字、GROUP BY 子句、ORDER BY 子句、HAVING 子句、UNION 运算符、子查询这些 SQL 语句结构，或者视图中的 FROM 子句中包含多个表、SELECT 子句中引用了不可更新视图，那么该视图都是不可更新的。

【例 5-7】 使用 INSERT 语句，通过视图 v_student 向学生信息表 S 中插入一条新的记录（'2020010','张三','男','21',计算机系）。

完成此操作的 SQL 语句如下。

```
mysql > INSERT INTO v_student
    - > VALUSE ('2020010','张三','男','21',计算机系);
Query OK, 0 rows affected (0.09 sec)
```

然后使用下列语句来查看学生信息表 S 中的数据情况。

```
mysql > SELECT * FROM student;
```

【例 5-8】 使用 UPDATE 语句，通过视图 v_student 更新学生信息表 S 中“年龄”列的所有取值为 20。

完成此操作的 SQL 语句如下。

```
mysql > UPDATE v_student
    - > SET Sage = 20;
Query OK, 0 rows affected (0.09 sec)
```

然后使用下列语句来查看学生信息表 S 中的数据情况。

```
mysql > SELECT * FROM S;
```

【例 5-9】 使用 DELETE 语句，通过视图 v_student 删除学生信息表 S 中学生张三的信息。

完成此操作的 SQL 语句如下。

```
mysql > DELETE FROM v_student
    - > WHERE Sname = '张三';
Query OK, 0 rows affected (0.09 sec)
```

然后使用下列语句来查看学生信息表 S 中的数据情况。

```
mysql > SELECT * FROM S;
```

6. 查询视图数据

视图被定义之后，就可以如同查询数据库中的表一样，对视图进行数据查询。查询操作是对视图使用最多的一种操作。例如，可以利用视图简化复杂的表连接，可以使用视图重新格式化检索出的数据，还可以使用视图过滤不想要的数据。

在 MySQL 中，对视图的使用还需要注意以下几点。

（1）创建视图必须具有足够的访问权限。

（2）对于可以创建的视图数目没有限制。

（3）视图可以嵌套。

（4）视图不能索引，也不能有关联的触发器、默认值。

（5）视图可以和基本表一起使用。

（6）每次使用视图时都必须处理查询执行时所需的检索操作。

5.3　审计跟踪和数据加密

5.3.1　审计跟踪

为了使数据库管理系统达到一定的安全级别，还需要根据安全策略提供其他方面的支持。审计功能就是数据库管理系统达到 C2 以上安全级别必不可少的一项指标。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，从而找出非法存取数据的人、时间和内容等。此外，也可以通过对审计日志的分析，对潜在的威胁提前采取措施加以防范。

可审计的事件有服务器事件、系统权限、语句事件及模式对象事件，还包括用户鉴别、自主访问控制和强制访问控制事件。

审计事件比较耗时，所以一般数据库管理系统将审计设置为可选特征，允许数据库管理员根据具体应用对安全性要求自主选择打开或关闭审计功能。

数据库审计主要用于监视并记录对数据库服务器的各类操作行为，并记入审计日志或数据库中以便日后进行跟踪、查询、分析，从而实现对用户操作的监控和审计。审计是一项非常重要的工作，也是企业数据安全体系的重要组成部分。

5.3.2　数据加密

数据库加密是指将存储于数据库中的数据，尤其是敏感数据，以加密的方式进行存储。数据是信息系统中最核心的部分，数据的丢失、破坏或泄漏，都会带来难以估量的损失。对敏感数据进行加密是数据安全防护中最核心的手段之一。

数据类型分为两种，一种是非结构化数据（例如文档和图片等），另一种是结构化数据

（例如数据库中的数据等）。这两种形态的数据都需要进行加密保护。加密后，数据以密文的方式存储，防止了数据直接暴露，同时增强了对加密数据的访问控制，大大降低了数据被泄漏和恶意破坏的风险。

由于国内所使用的数据库管理系统大都是国外产品，出于安全可控的考虑，需要国产的数据库加密产品。

国产数据库加密产品的发展可以分为以下三个阶段。

第一个阶段是摸索阶段。

在2003年之前，国内的数据库加密手段是通过反编译国外安全数据库系统完成。国外安全版本的数据库系统具有加密功能，由国内技术人员对其进行逆向工程，加入国产加密算法，完成“国产化”。

第二阶段是国外产品导入国内市场以及国产数据库加密产品萌芽阶段。

从2003年开始，国外的数据库加密产品厂商，为了进入中国市场，将产品界面进行“中国化”，经由中国香港进入中国内地市场。但由于国家保密政策的限制，这些被伪装成国产的数据库加密产品并没有在国内数据库安全市场大行其道，反而逐渐销声匿迹。

在这一阶段，逐渐有国内科研人员开始进行数据库加密技术的研究。2009年已有数据库加密技术的发明专利出现。随后，国内陆续有研发团队开始进行数据库加密产品的开发，迈出了非常重要的第一步。

第三个阶段是国产数据库加密技术逐步产品化并走向市场的阶段。

从2010年开始，随着科研成果的产业化，国内市场开始出现国产的数据库加密产品。经过市场的磨练，产品越来越成熟，越来越为数据库安全运维人员所接受。

数据库加密的实现方式主要有全盘加密、文件加密、数据库自带加密、库内扩展加密、数据库加密网关和应用加密网关。

全盘加密是指采用全盘加密系统或者存储加密网关系统，将数据库文件所在的磁盘扇区进行加密。当数据库访问磁盘扇区的时候，对加密扇区再进行解密。这种方式对于数据库自身来说是透明的，数据库管理系统也感觉不到加密解密过程的存在。这种加密方式工作在存储层，仅能防止磁盘丢失时敏感数据遭受泄漏。所有对磁盘具有访问权限的用户都可以访问到真实的数据库文件，因此对于控制了操作系统的攻击者来说没有防护能力。

文件加密是指在操作系统文件驱动层将数据库的存储文件经过加密后存储到磁盘上。当数据库访问存储文件的时候，再进行解密。这种方式对于数据库自身来说也是透明的，数据库管理系统也感觉不到加密解密过程的存在。这种加密方式能防止磁盘丢失和文件被复制导致的敏感数据泄漏。但是，对于控制了数据库系统的攻击者来说，文件还是开放的，因此也没有真正的防护能力。

数据库自带加密是指某些数据库自身提供了加密机制，在数据库内核实现了存储的加密。这种加密方式能防止磁盘丢失和文件被复制导致的敏感数据泄漏。但是，对于控制了数据库系统的攻击者来说却是开放的，并没有防护能力，而且其密钥管理通常不会对数据库用户开放，安全性得不到保证。

库内扩展加密是指通过使用视图、触发器、扩展索引等机制，实现透明加密。由于引入了独立于数据库的第三方程序，通过控制加密解密的权限，增加了额外的访问控制。对于数据库内不同的用户，也可以控制其对加密数据的访问。但是这种加密方式不能越过应用系

统，并且依赖于数据库系统的扩展索引机制，并不能在所有数据库上实现。

数据库加密网关或加密驱动是指通过对数据库前端部署数据库加密网关，或者通过扩展数据库访问驱动（如 JDBC 驱动）实现数据库加密。这种方式理论上能够支持所有的数据库，是一种通用的解决方案，且安全性更高。但是对于所有访问语句和访问机制却难以全部支持，例如，对于网关之后的存储过程和触发器都无法支持。

应用加密网关是指在应用系统之前放置加密网关，进一步将数据加密的位置提前，在数据进入应用系统之前进行加密。这种加密方式可以控制应用系统的用户对数据的访问权限，并且真实数据对所有数据库用户都是不可见的，是最安全的一种加密方式。事实上，这种加密方式与具体的数据库无关，独立于数据库。但是由于应用系统的复杂性，实现的难度也较大。

总之，数据被加密的位置离用户越近，安全性越高，同时实现的难度也越大。以上所述的几种加密方式，数据加密的位置离用户是逐步靠近的，防护能力也逐步提升。

一般数据库加密能够通过有效地解决如下问题，来提升数据库的安全性。

（1）防止数据库文件被下载或者复制，以及直接分析数据文件导致的数据泄露和破坏。

（2）防止 DBA 或高权限账号密码泄露导致的数据泄露和破坏。

（3）实现多因子身份认证和授权，弥补仅由口令验证方式安全性不足的缺陷。

小结

随着信息时代的发展，数据库应用技术不断深入，数据及数据库的安全性越来越重要。数据库管理系统在数据的管理和控制方面起到核心的作用，必须具有完整而有效的数据库安全机制。

本章讨论了数据库安全性控制机制的一般理论，包括用户标识与鉴别、存取控制策略、自主存取控制、强制存取控制和视图机制；后续章节将会以 MySQL 数据库管理系统为例，讨论其安全机制。

本章最后概述了审计跟踪技术和数据加密技术，为有兴趣的读者学习起到抛砖引玉的作用。

习题

1. 概念题

（1）数据库安全性和计算机系统的安全性有什么关系？

（2）简述数据库安全性控制的常用方法和技术。

（3）SQL 中提供了哪些数据控制语句？请举例说明它们的使用方法。

（4）请分析数据库的审计功能，并说明数据库提供审计功能的重要性。

（5）请分析统计数据库安全性问题具有何种特殊性。

2. 操作题

（1）使用 MySQL 数据库管理系统，创建多个新用户，并对这些用户进行管理，分配相应的数据库操作权限。

（2）使用 MySQL 数据库管理系统，基于当前数据库表创建不同的视图，并为（1）中创建的用户分配这些视图的操作权限。