|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| 数据库系统及安全  实验手册 | | | |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  | **学号：20373108** | |  |
|  |  |  |  |
|  | **姓名：张耀东** | |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **网络空间安全学院**  **2022年秋季** | | | |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

实验四、数据库系统安全实验

1. 实验目的
2. 熟悉与掌握数据库安全的基本概念和技术；
3. 了解数据库安全的身份认证与访问控制机制及其使用方法；
4. 了解数据库管理系统安全加固的基本方法；
5. 掌握数据库备份与恢复的基本方法；
6. 了解SQL注入的基本原理和预防方法。
7. 实验要求
8. 本次实验为个人独立实验，作业模板可基于本实验手册，在实验内容后面直接填写实验报告。
9. 作业提交方式：电子版(rar/zip打包)，文件命名格式：[学号]-[姓名]-实验[序号1位数，1--9].rar/zip；作业文件建议使用word或者pdf格式，不接受拍照图片版本。
10. 作业提交到北航网盘：共享目录“DBMS-2022-作业”上传

https://bhpan.buaa.edu.cn:443/link/53C3AF672185198F3381B8E0F83229DF 密码：z0sD

1. 每一个实验内容，根据要求，进行实际操作，并把具体的步骤记录下来，如给出数据操作等SQL语句，结果截图后附在后面。
2. 实验内容
3. 针对第三次实验课选择的实际应用场景，对应用程序访问数据库的用户设计合理的数据库访问权限，基于“最小权限”的原则，要求如下(基于Oracle)：
   1. 不能具有DBA的权限
   2. 不能创建、修改（表结构）和删除（整张表）任何表
   3. 基于应用系统的数据访问需求设置相应的权限
   4. 给出上述所有授权的sql脚本，并作基本验证（结果截图）

|  |  |
| --- | --- |
| 创建名为user1的用户，密码为user1 | create user user1 identified by user1; |
| 赋予user1登录和访问表空间的权限 | grant create session to user1;  grant unlimited tablespace to user1; |
| 根据实验3实际情况，授权用户（包括管理员用户）对三张表格进行查询、插入、更新操作。 | grant select on Students to user1;  grant select on Classifications to user1;  grant select on Borrows to user1;  grant select on Books to user1;  grant insert on Students to user1;  grant insert on Classifications to user1;  grant insert on Borrows to user1;  grant insert on Books to user1;  grant update on Students to user1;  grant update on Classifications to user1;  grant update on Borrows to user1;  grant update on Books to user1; |
| SQL脚本截图 |  |
| 授权截图 |  |
| 权限验证截图 |  |
| 权限受限验证 |  |

1. 设计一个应用程序访问数据库所使用的用户名和密码（不能直接写在代码里，建议使用配置文件）存储与解析的解决方案，解决明文存储问题，给出详细的方案描述。

|  |  |
| --- | --- |
| 方案设计 | 利用已有的加解密工具，实现本地对用户名和密码的加解密。在配置文件里填写用户名与密码的密文，并写好接口，使得在登录数据库时将密文传输到到本地加解密应用程序，经过个人确认来对传来的密文（用户名和密码的密文）后进行解密然后回传，从而连接数据库。 |
| 现有方案 | 加密工具：Druid中的工具类com.alibaba.druid.filter.config.ConfigTools，使用其中的encrypt(String plainText)方法和decrypt(String cipherText)方法采用默认的公私玥加解密。  加密文件配置：    重写用户名和密码的方法为解密后端登录时传入的数据：    在Spring文件中配置数据源：    由此重写了后端登录接口处的输入。 |

1. 针对Oracle数据库，对其进行基本安全加固，要求如下：
   1. 在安装数据库的机器上，要求任何用户登录数据库必须提供密码，不可以dba直接登录数据库(如 sqlplus / as sysdba)；
   2. 限制用户密码的复杂度（包括字母、数字和特殊字符等），设置每隔60天需要重新修改密码，如果用户密码连续输错3次就锁定用户（不能再登录），设置用户登录session空闲超时间隔为10分钟。
   3. 限制应用程序从某些固定的IP地址访问数据库服务器。

|  |  |
| --- | --- |
| 1.将<oracle安装路径\homes\OraDB21Home1\network\admin\sqlnet.ora> 文件中SQLNET.AUTHENTICATION\_SERVICES= (NTS)行进行注释，即设置为口令文件认证模式：oracle认为操作系统是不可靠的，若要访问数据库，必须要输入用户密码，不可以dba直接登录数据库。 |  |
| 2.首先创建profile，对资源进行限定，然后创建用户，将profile应用于用户User1 | 其中，口令复杂度验证如下： |
| 3.将<oracle安装路径\homes\OraDB21Home1\network\admin\sqlnet.ora>添加最后两行：<tcp.validnode\_checking=yes>（开启ip限制功能），<tcp.invited\_nodes=(127.0.0.1)> （只允许127.0.0.1的ip访问） |  |

1. 对某一使用的数据库（可以是第一次实验的销售数据库，也可以是第三次实验的应用场景数据库），采用当前的用户对数据库逻辑数据备份与恢复，然后再使用与备份数据库不一样的用户名进行数据恢复(使用Oracle IMPDP/EXPDP进行备份与恢复)，给出执行结果与验证截图。（提示：为了进行数据恢复，先做基于schema的数据备份，然后drop当前用户，再新建用户，再导入数据。 如果用户名、表空间名称不一致，可以使用REMAP\_SCHEMA、REMAP\_TABLESPACE选择项进行映射）

|  |  |
| --- | --- |
| 指令 | -- 到xepdb1会话中  alter session set container = xepdb1;  -- 创建用户  create user OT identified by ot;  -- 赋予用户权限  GRANT CONNECT, RESOURCE, DBA TO OT;  -- 导入初始数据  @./oracle\_sql/schema\_oracle.sql;  @./oracle\_sql/data\_oracle.sql;  -- 创建directory对象  create directory backup as 'D:\TmpFile\db\_bk\';  -- drop directory backup;  -- 将当前目录的权限授予  grant read,write on directory backup to public;  -- 退出  exit;  -- 以下在命令行（cmd）中执行  -- 导出  expdp OT/ot@xepdb1 tables = Students job\_name=ex4\_data1 directory=backup parallel=1 dumpfile=Students.dmp content=all logfile=ex4\_data1.log  expdp OT/ot@xepdb1 tables = Classifications job\_name=ex4\_data4 directory=backup parallel=1 dumpfile=Classifications.dmp content=all logfile=ex4\_data4.log  expdp OT/ot@xepdb1 tables = Books job\_name=ex4\_data2 directory=backup parallel=1 dumpfile=Books.dmp content=all logfile=ex4\_data2.log  expdp OT/ot@xepdb1 tables = Borrows job\_name=ex4\_data3 directory=backup parallel=1 dumpfile=Borrows.dmp content=all logfile=ex4\_data3.log  -- 到xepdb1会话中  alter session set container = xepdb1;  -- 删除用户  drop user OT CASCADE;  -- 创建新用户  create user ot\_bk identified by ot\_bk;  -- drop user ot\_bk CASCADE;  -- 赋予新用户权限  GRANT CONNECT, RESOURCE, DBA TO ot\_bk;  -- 导入  impdp ot\_bk/ot\_bk@xepdb1 tables="OT"."STUDENTS" directory=backup dumpfile=Students.dmp REMAP\_SCHEMA=OT:ot\_bk REMAP\_TABLESPACE=OT:ot\_bk  impdp ot\_bk/ot\_bk@xepdb1 tables="OT"."Classifications" directory=backup dumpfile=Classifications.dmp REMAP\_SCHEMA=OT:ot\_bk REMAP\_TABLESPACE=OT:ot\_bk  impdp ot\_bk/ot\_bk@xepdb1 tables="OT"."Books" directory=backup dumpfile=Books.dmp REMAP\_SCHEMA=OT:ot\_bk REMAP\_TABLESPACE=OT:ot\_bk  impdp ot\_bk/ot\_bk@xepdb1 tables="OT"."Borrows" directory=backup dumpfile=Borrows.dmp REMAP\_SCHEMA=OT:ot\_bk REMAP\_TABLESPACE=OT:ot\_bk |
| 截图验证 | 首先在xepdb1下创建OT并授权，并建立四张表（schema为默认）  然后创建目录，名为backup，并对公共开放（或者分别对OT和之后建立的ot\_bk开放，这里简写为public）：      输入上述指令进行备份并导出：          backup所在文件下导出后显示：    在sysdba视角下删除OT，并建立新的用户ot\_bk：    利用ot\_bk身份导出数据库内的表（默认schema）：          验证导出数据库是否可用： |

1. SQL注入验证：针对实验三开发的应用程序，验证SQL注入，并提供相应的解决方案，预防SQL注入；要求至少验证2个不同的SQL注入场景。给出测试场景、测试过程、测试结果、预防方案与实际结果。

|  |  |
| --- | --- |
| 在实验三中对sql注入已有防备，主要利用？进行传参操作，？默认对参数进行编码，具有防止sql注入的功能，仅有的三处使用${}进行传参的功能分别在书籍查询（所有人）、借阅记录查询（管理员）、修改书名（管理员）处，皆是由于为了使用模糊比较而寻求方便才使用而${}。下面针对前两处进行讨论，并给出两种解决方案。 | |
| 注入场景一：在图书搜索栏中sql语句为<**SELECT** books.book\_name,books.author\_name,books.ISBN,classifications.classification,classifications.location,books.**current**,books.total **FROM** books,classifications **WHERE** books.classification = classifications.id **AND** books.book\_name **LIKE** '%${data.info}%';>，利用${}进行传参（将传入的数据直接显示生成在sql语句中），使得  可以导入适当的sql语句使得执行结果不同，如：<冯%' union SELECT books.book\_name,books.author\_name,books.ISBN,classifications.classification,classifications.location,books.current,books.total FROM books,classifications WHERE books.classification = classifications.id AND books.book\_name LIKE '%老人与海>  查询结果包含了“冯”与“老人与海”两个关键词    同理，可以利用sql注入在没有管理员权限的情况下查询其他用户信息 | 解决方案：利用mysql.escape（）对传入参数进行编码，防止篡改sql语句使其发挥其他作用。    修改后语句为<**SELECT** books.book\_name,books.author\_name,books.ISBN,classifications.classification,classifications.location,books.**current**,books.total **FROM** books,classifications **WHERE** books.classification = classifications.id **AND** books.book\_name **LIKE** ${mysql.**escape**("%"+**data**.info+"%")};>  修改输入尝试sql注入：<冯%' union SELECT books.book\_name,books.author\_name,books.ISBN,classifications.classification,classifications.location,books.current,books.total FROM books,classifications WHERE books.classification = classifications.id AND books.book\_name LIKE '%老人与海  >  注入结果： |
| 注入场景二：在查询借阅记录中使用规定sql语句为<**SELECT** borrows.id,students.Sno,students.**name**,books.book\_name,books.author\_name,books.ISBN,borrows.borrow\_date,**DATE\_ADD**(borrows.borrow\_date,**INTERVAL** 30 **day**)date\_time,borrows.return\_date,**if**(borrows.return\_date **IS** **NULL**,'未还','已还') **status** **FROM** borrows,students,books **WHERE** borrows.Sno=students.Sno **AND** borrows.ISBN=books.ISBN **AND** books.book\_name **LIKE** '%${data.info}%';>  设置输入参数为<老人%';DELETE FROM borrows WHERE sno != '0>      数据库已经被删除 | 解决方案：利用?引入参数会使得引入值进行编码，防止篡改sql语句使其发挥其他作用。    修改后语句为<**SELECT** borrows.id,students.Sno,students.**name**,books.book\_name,books.author\_name,books.ISBN,borrows.borrow\_date,**DATE\_ADD**(borrows.borrow\_date,**INTERVAL** 30 **day**)date\_time,borrows.return\_date,**if**(borrows.return\_date **IS** **NULL**,'未还','已还') **status** **FROM** borrows,students,books **WHERE** borrows.Sno=students.Sno **AND** borrows.ISBN=books.ISBN **AND** books.book\_name **LIKE** ?;>  修改输入尝试sql注入：<老人%';DELETE FROM borrows WHERE sno != '0>  注入结果：失败    刷新后：    数据库内显示： |

1. 实验总结与建议：针对本学期的课程实验，做实验总结（如收获、经验或不足等），并对本实验课程给出相应的意见或者建议。

|  |
| --- |
| 答：本学期通过数据库实验能够加深我对老师上课传授的知识的理解，并且切实学到许许多多的东西。如从第一节课开始对oracle比较陌生，到第二节课可以较为熟练地进行增删改查数据，建立用户和表等等。在第三节课，更是促使我学习了网页前后端的开发以及后端与数据库的连接。在这第四节课切实主题，交了许多关于数据库安全的东西，也让我学会了许许多多重要的知识点，让我意识到之前写的系统虽然功能完备，但是在安全方面很可能因为sql注入而遭受入侵，或者后端对数据库端进行连接时的配置文件使用明文更是十分危险。当然数据库实验还锻炼了我的能力：根据下发的资料和网络，理解并实现需要的功能，其中由于oracle版本不同经常会出现许许多多意想不到的问题。以这次实验为例（这次试验印象还很深刻），在进行数据库备份和导入时，即使使用了REMAP\_TABLESPACE与REMAP\_SCHEMA也依旧显示操作错误，无法找到对象，从网上找到的指令写法并无太大差异，最后翻阅backup中的数据导出log中发现是因为命名规范的不同：导出的table默认为”OT”.”table”（原用户名：OT，表名：table），网上的写法一般为OT.table或’OT.table’，而当我改为”OT”.”table”时才成功。类似的示例还有很多。总之数据库实验这门课虽然学分低，但是要求并不低，我也确确实实在这门课中学到了许多东西。 |