1. 查询没有考试成绩的学生姓名和课程名 (1)

```
SELECT SName, CName
FROM GRADE JOIN STUDENT JOIN COURSE
WHERE score IS NULL;
```

2. 查询和汪远在同一个系学习的学生姓名，宿舍号和电话 (1)

```
SELECT SNAME, S.DORMN, TELE
FROM STDUDENT S JOIN DORM
WHERE DNO = (
    SELECT DNO
    FROM STUDENT
    WHERE SNAME = '汪远'
);
```

3. 查询选修了1号课程的所有姓张的同学的姓名和1号课程的成绩(1)

```
SELECT SNAME, SCORE
FROM STUDENT JOIN GRADE
WHERE CNO = '01' AND SNAME = '张%';
```

4. 查询数学系所有学生的住宿情况，生成如下结果：(1)

```
姓名      宿舍号    宿舍电话


XXX      XXX      XXX
```

```
SELECT SNAME 姓名， D.DORMN 宿舍号， TELE 宿舍电话
FROM STUDENT S JOIN DORM D
WHERE DNO = (
    SELECT DNO
    FROM DEPARTMENT
    WHERE DNAME = '数学'
);
```

5. 查询数据库原理这门课成绩不低于80分的学生姓名和成绩 (1)

```
SELECT SNAME， SCORE
FROM STUDENT JOIN GRADE
WHERE SCODE > 80;
```

6. 查询与‘原野’同岁的学生姓名（不包括原野本人）(1)

```
SELECT SNAME
FROM STUDENT
WHERE NAME <> '原野' AND AGE = (
    SELECT AGE
    FROM STUDENT S
    WHERE S.SNAME = '原野'
);
```

7. 查询每门课中成绩最低的学生学号、课程号、成绩，并按课程号排序。

```
SELECT SNO, CNO, GRADE
FROM STUDENT S AND GRADE G
WHERE EXISTS (
    SELECT *
    FROM GRADE G1
    WHERE G1.CNO = S.CNO AND S.SNO = G.SNO AND
G.SCORE = (
        SELECT MIN(SCORE)
        FROM GRADE G2
        WHERE G2.CNO = G.CNO
    )
);
```

8. 查询'王凯'老师带的课程名和学生人数(1)

```
SELECT CNAME, COUNT(g.SNO)
FROM GRADE g JOIN COURSE
GROUP BY CNO;
```

9. 查询给'计算机系'学生讲课的老师(1)

```
SELECT DISTINCT COURSE.TEACHER
FROM COURSE, STUDENT, DEPARTMENT
WHERE COURSE.CPno = DEPARTMENT.DNo AND
STUDENT.DNo = DEPARTMENT.DNo AND
DEPARTMENT.DName = '计算机系';
```

10. 查询宿舍电话是8302202的学生学号和姓名 (1)

```
SELECT SNAME, SNO
FROM STUDENT JOIN DROM
WHERE TELE = '8302202';
```

11. 查询选修了没有先行课的课程的学生学号和姓名 (1)

```
SELECT SNO, SNAME
FROM STUDENT JOIN GRADE G
WHERE G.CNO IN (
    SELECT G1.CNO
    FROM COURSE
    WHERE CPNO IS NULL
)
GROUP BY SNO;
```

12. 查询选修了学分是4分的课程的学生学号、姓名、性别和年龄，结果按性别升序、年龄降序排序 (1)

```
SELECT SNO, SNAME, SEX, AGE
FROM STUDENT S JOIN GRADE G
WHERE G.CNO IN (
    SELECT C.CNO
    FROM COURSE C
    WHERE C.CREDIT = 4
)
ORDER BY SEX ASC, AGE DESC;
```

13. 查询数学系、物理系、中文系的学生情况（要求：用三种方法实现查询）(1)

```
--1.
SELECT *
FROM STUDENT JOIN DEPARTMENT D
WHERE D.DNAME IN {'数学系','物理系','中文系'};
--2.
SELECT *
FROM STUDENT
WHERE DNO IN (
    SELECT DNO
    FROM DEPARTMENT
    WHERE DNAME IN {'数学系','物理系','中文系'}
);
--3.
SELECT *
FROM STUDENT JOIN DEPARTMENT D
WHERE D.DNAME = '数学系' OR D.DNAME = '物理系' OR
D.DNAME = '中文系';
```

14. 查询所有课程的后继课程，生成如下结果：（要求：按照课程号排序）(1)

    课程号　后继课程号

    XXX　　XXX

    ```
    SELECT CNO AS 课程号, CPNO AS 课程号
    FROM COURSE
    ORDER BY CNO;
    ```

15. 查询每个宿舍中成绩最高的学生的宿舍号、学号和姓名 (1)

    ```
    SELECT DORMNO, SNO, SNAME
    FROM STUDENT S JOIN GRADE G
    HAVING MAX(SCORE) > ALL(
        SELECT SCORE
        FROM STUDENT S1 JOIN GRADE G1
        WHERE S1.DORMNO = S.DORMNO
    )
    GROUP BY SNO;
    ```

16. 查询计算机系所有1985年出生的学生（以2005年为标准）(1)

    ```
    SELECT SNO, SNAME
    FROM STUDENT
    WHERE AGE = 2005-1985;
    ```

17. 查询1系选修了02号课程的最高分、最低分、平均分 (1)

```
SELECT MAX(SCORE), MIN(SCORE), AVG(SCORE)
FROM STUDENT JOIN SCORE
WHERE DNO = '1' AND CNO = '02';
```

18. 查询所有不及格同学的基本情况 (1)

```
SELECT *
FROM STUDENT JOIN GRADE
WHERE SCORE < 60;
```

19. 查询GRADE表中成绩在60到80分之间的所有成绩记录信息；(1)

```
SELECT *
FROM GRADE
WHERE SCORE BETWEEN 60 AND 80;
```

20. 查询中文系年龄在20岁以下的学生姓名。(1)

```sql
SELECT SNAME
FROM STUDENT
WHERE AGE < 20 AND DNO = (
    SELECT DNO
    WHERE DNAME = '中文系'
);
```

21. 查询所有”01”系或性别为”女”的同学记录；(1)

```sql
SELECT *
FROM STUDENT
WHERE DNO = '01' OR SEX = '女';
```

22. 查询王老师讲的每门课的学生平均成绩，输出课程号和平均成绩。(1)

```sql
SELECT CNO, AVG(SCORE)
SELECT GRADE G
WHERE G.CNO IN (
    SELECT C.CNO
    FROM COURSE C
    WHERE C.TEACHER = '王老师'
);
```

23. 查询所有男学生情况并按年龄升序排。(1)

```
SELECT *
FROM STUDENT
WHERE SEX = '男'
ORDER BY AGE ASC;
```

24. 查询选修了课程’01’但没有选修课程’02’的学生学号(1)

```
SELECT SNO
FROM STUDENT
WHERE SNO IN (
    SELECT SNO
    FROM GRADE
    WHERE CNO = '01'
) AND SNO NOT IN (
    SELECT SNO
    FROM GRADE
    WHERE CNO = '02'
);
```

25. 查询’物理系’学生的学号、姓名和宿舍情况，结果按宿舍号升序排列。(1)

```
SELECT SNO, SNAME, DROMNO
FROM STUDENT
WHERE DNO = (
    SELECT DNO
    FROM DEPRTMENT
    WHERE DNAME = '物理系'
);
```

26. 查询‘中文’系学生的详细记录情况，结果按性别升序、年龄降序排列。(1)

```
SELECT *
FROM STUDENT
WHERE DNO = (
    SELECT DNO
    FROM DEPRTMENT
    WHERE DNAME = '中文系'
)
ORDER BY SEX ASC, AGE DESC;
```

27. 查询‘2’系的姓名和其出生年月(以2005年为标准)，并用“BIRTHDAY”改变结果标题。(1)

```
SELECT SNAME, 2005-AGE AS BIRTHDAY
FROM STUDENT
WHERE DNO = '2';
```

28. 查询所有被选修的课程情况。(1)

```
SELECT *
FROM COURSE C
WHERE C.CNO IN (
    SELECT G.CNO
    FROM GRADE
    GROUP BY CNO
);
```

29. 查询年龄在21-25之间（包括21和25）的学生姓名，宿舍号，电话。(1)

```
SELECT SNAME, DROMNO, TELE
FROM STUDENT JOIN DORM
WHERE (2005 - SAGE) BETWEEN 21 AND 25;
```

30. 查询‘计算机’系、‘中文’系和‘物理’系的学生姓名，宿舍号。(1)

```
SELECT SNAME, DORMNO
FROM STUDENT
WHERE DNO IN (
    SELECT DNO
    FROM DEPRTMENT
    WHERE DNAME IN {'物理系','中文系','物理系'}
);
```

31. 查询所有姓‘张’的学生情况。(1)

```sql
SELECT *
FROM STUDENT
WHERE SNAME LIKE '张%'
```

32. 查询COURSE表中有先修课程的课程名和教师情况 (1)

```sql
SELECT CNAME, TEACHER
FROM COURSE
WHERE CNO IN (
    SELECT CNO
    FROM COURSE
    WHERE CPNO IS NOT NULL
);
```

33. 查询‘计算机’系的所有男学生的信息 (1)

```sql
SELECT *
FROM STUDET
WHERE SEX = '男' AND DNO = (
    SELECT DNO
    FROM DEPRTMENT
    WHERE DNAME = '计算机系'
);
```

34. 查询所有‘2’系的学生人数。 (1)

```
SELECT COUNT(*)
FROM SUTDENT
WHERE DNO = '2';
```

35. 查询所有已选修课程的学生的学号、姓名、选修的课程名和授课教师信息。(1)

```
SELECT SNO, SNAME, CNAME, TEACHER
FROM STUDENT JOIN GRADE JOIN COURSE;
```

36. 查询是系主任‘李永军’的学生的姓名和宿舍电话。 (1)

```
SELECT SNAME, TELE
FROM STUDENT
WHERE DNO = (
    SELECT DNO
    FROM DEPRTMENT
    WHERE HEAD = '李永军'
);
```

37. 查询其他系比‘中文系’所有学生年龄大的学生姓名和年龄。(1)

```
SELECT SNAME, SAGE
FROM STUDENT
WHERE SAGE >  (
    SELECT MAX(SAGE)
    FROM STUDENT
    WHERE DNO = (
        SELECT DNO
        FROM DEPARTMENT
        WHERE DNAME = '中文系'
    )
);
```

38. 查询选修课程库没有不及格分数的学生。(1)

```
SELECT *
FROM STUDENT S
WHERE NOT EXISTS (
    SELECT G.CNO
    FROM GRADE G
    WHERE G.SNO = S.SNO AND SCORE < 60
);
```

39. 查询'3'系的学生与年龄大于19岁的学生的差集。(1)

```sql
SELECT *
FROM STUDENT
WHERE DNO = '3'
EXCEPT
SELECT *
FROM STUDENT
WHERE AGE > 19;
```

40. 查询选修“数据库原理”的学生与选修“C语言”的学生的交集。
    (1)

```sql
SELECT *
FROM STDENT
WHERE SNO IN (
    SELECT G.SNO
    FROM GRADE
    WHERE G.CNO = (
        SELECT C.CNO
        FROM COURSE
        WHERE C.CNAME = '数据库原理'
    )
)
INTERSECT
SELECT *
FROM STDENT
WHERE SNO IN (
    SELECT G.SNO
    FROM GRADE
    WHERE G.CNO = (
        SELECT C.CNO
        FROM COURSE
        WHERE C.CNAME = 'C语言'
    )
);
```

41. 从学生选课关系SC中，删除李军（学生关系中可能有重名）的所有选课(1)

```
DELETE FROM SC
WHERE SNo IN (
    SELECT SNo
    FROM STUDENT
    WHERE SName = '李军'
);
```

42. 对STUDENT表以学号SNO，创建聚簇索引。（1）

```
CREATE CLUSTERED INDEX idx_sno ON STUDENT(SNO);
```

43. 对STUDENT表以学号DORMNO，创建唯一索引。（1）

```
CREATE UNIQUE INDEX idx_dormno ON
STUDENT(DORMNO);
```

44. 将计算机系学生选修课程中，成绩为空的学生成绩置零 (1)

```sql
UPDATE SC
SET Score = 0
WHERE CNo IN (
    SELECT CNo
    FROM COURSE
    WHERE Dept = '计算机系'
) AND Score IS NULL;
```

45. 对所有“计算机系”的学生的选修课程的成绩提高10%。（1）

```sql
UPDATE GRADE
SET Score = Score * 1.1
WHERE CNo IN (
    SELECT CNo
    FROM COURSE
    WHERE Dept = '计算机系'
);
```

46. 将COURSE课程信息表中有先修课的课程的学分增加2分。(1)

```sql
UPDATE COURSE
SET CREDIT = CREDIT + 2
WHERE CPNO IS NOT NULL;
```

47. 将student中学号为”990303”的学生的系号改为“3”；（1）

```
UPDATE STUDENT
SET DNO = '3'
WHERE SNO = '990303';
```

48. 插入一条学生记录（960101，张华，女，21，4，2303）。(1)

```
INSERT INTO STUDENT(SNO，SNAME,
SEX,AGE,DNO,DROMNO)
VALUE('960101'，'张华'，'女'，21，'4'，'2303')
```

49. 创建计算机系所有不及格学生的视图 (1)

```
CREATE VIEW student_not_passed
AS
SELECT SNAME，SNO
WHERE DNO = (
    SELECT DNO
    FROM DEPARTMENT
    WHERE DNAME = '计算机系'
) AND SNO IN (
    SELECT SNO
    FROM GRADE
    WHERE SCORE < 60
);
```

50. 创建“计算机”系的所有男生的视图VIEW_1（要求反映出学生的出生年份）。(1)

```
CREATE VIEW VIEW_1
AS
SELECT SNO,SNAME,(2022-AGE) AS BIRTHDAY
WHERE SEX = '男' AND DNO = (
    SELECT DNO
    FROM DEPRTMENT
    WHERE DNAME = '计算机系'
);
```

51. 修改视图VIEW_1中的学生“明天”的年龄为23，宿舍号为“2202”。(1)

```
UPDATE VIEW_1
SET SAGE = 23, DORMNO = '2202'
WHERE SNAME = '明天';
```

52. 把用户U2修改学生学号的权限收回。 （以上操作，需要检查是否授权成功。）(1)

```
REVOKE UPDATE(SNO) ON STUDENT FROM U2;
SHOW GRANTS FOR U2;
```

53. 对表STUDENT的INSERT权限授予U1用户，并允许他再将此权限授予其他用户。 （以上操作，需要检查是否授权成功。）(1)

```
GRANT INSERT ON STUDENT TO U1 WITH GRANT OPTION;
SHOW GRANTS FOR U1;
```

54. 创建一个“TEST”用户，默认数据库为STUDENT_DATA(应用实例中数据环境)，且享有一切数据库操作权限。 （以上操作，需要检查是否授权成功。）(1)

```
CREATE LOGIN TEST WITH PASSWORD = 'password',
DEFAULT_DATABASE = STUDENT_DATA;
CREATE USER TEST FOR LOGIN TEST;
USE STUDENT_DATA;
GRANT CONTROL ON DATABASE::STUDENT_DATA TO TEST;
```

55. 将“TEST”用户修改学生姓名的权限收回。 （以上操作，需要检查是否授权成功。）(1)

```
REVOKE UPDATE(SNAME) ON STUDENT FROM TEST;
SHOW GRANTS FOR TEST;
```

56. 收回“TEST”用户对STUDENT表的所有权限。 （以上操作，需要检查是否授权成功。）

```
REVOKE ALL PRIVILEGES ON STUDENT FROM TEST;
SHOW GRANTS FOR TEST;
```

57. 将对GRADE表的查询权限授给“TEST”用户。（以上操作，需要检查是否授权成功。）(1)

```
GRANT SELECT ON GRADE TO TEST;
SHOW GRANTS FOR TEST;
```

58. 查询每个宿舍住宿的人数（要求列出宿舍号、宿舍人数）（2）

```
SELECT DORMNO,COUNT(SNO)
FROM STUDENT
GROUP BY DORMNO;
```

59. 查询每个系主任所在系的学生人数（要求列出系主任名称、所在系名和系中学生人数）（2）

```
SELECT HEAD， DNAME， COUNT(SNO)
FROM STUDENT JOIN DEPARTMENTS
GROUP BY DNO;
```

60. 查询计算机系总成绩最高的人，生成如下结果：（2）

姓名　总成绩

XXX　XXX

```
SELECT SNAME, SUM(SCORE)
FROM STUDENT JOIN GRADE
WHERE DNO = (
    SELECT DNO
    FROM DEPARTMENT
    WHERE DNAME = '计算机系';
)
HAVING SUM(SCORE) = (
    SELECT MAX(S)
    FROM (
        SELECT SUM(SCORE) S
        FROM GRADE
        GROUP BY SNO
    )
)
GROUP BY SNO;
```

61. 求所有系的学生平均成绩，并把结果存入新表Dep_avg_s（DNO, AVG_S）中，再从这个表中查询出所有结果（要求：如果某个系没有平均成绩，就将平均成绩置为空值）(2)

```
CREATE TABLE DEP_AVG_S
AS
SELECT DNO, AVG(sCORE) AS AVG_S
FROM STUDENT JOIN GRADE
GROUP BY SNO;
```

62. 查询其他系中比计算机系某一学生年龄小的学生姓名和年龄 (2)

```sql
SELECT SNAME, SAGE
FROM SUDENT
WHERE DNO <> (
    SELECT DNO
    FROM DEPARTMENT
    WHERE DNAME = '计算机系'
) AND SAGE < ANY (
    SELECT SAGE
    FROM STUDENT
    WHERE DNO = (
        SELECT DNO
        FROM DEPARTMENT
        WHERE DNAME = '计算机系'
    )
);
```

63. 查询选修课程中，有课程没有成绩、但是其他课程的成绩均在80分以上的同学的姓名、课程号、成绩 (2)

```
SELECT SNAME, CNO, SCORE
FROM STUDENT S JOIN GRADE G
WHERE  G.SCORE IS NULL AND (
    SELECT COUNT(*)
    FROM GRADE G1
    WHERE S.SNO = G1.SNO AND G1.GRADE > 80
) = (
    SELECT COUNT(*)
    FROM GRADE G1
    WHERE S.SNO = G1.SNO AND G1.CNO <> G.CNO
);
```

64. 查询选修课程总学分最高的学生姓名和总学分 (2)

```
SELECT TOP 1 SNAME, SUM(CREDIT)
FROM  STUDENT JOIN GRADE JOIN COURSE
GROUP BY SNO
ORDER BY SUM(CIRTICAL) DESC;
```

65. 查询显示所有学生选修课程的学分的累计情况，并按照总学分的高低顺序排序。 (2)

```
SELECT SNO, SNAME, SUM(CREDIT)
FROM STUDENT JOIN GRADE JOIN COURSE
GROUP BY SNO
ORDER BY SUM(CREDIT);
```

66. 查询所有未选修 ‘03’ 号课程的学生姓名(用存在量词)。(2)

```
SELECT SNAME
FROM STUDENT S
WHERE NOT EXISTS(
    SELECT *
    FROM GRADE G
    WHERE S.SNO = G.SNO AND CNO = '03'
);
```

67. 查询选修了课程名为’高等数学’的学生学号和姓名（用嵌套/用连接）。(2)

```
SELECT SNO, SNAME
FROM STUDENT S
WHERE SNO IN (
    SELECT SNO
    FROM GRADE G
    WHERE CNO = (
        SELECT CNO
        FROM COURSE
        WHERE CNAME = '高等数学'
    )
);
```

68. 创建数据表Grade1，将“99”级学生选修“高等数学”的的学生插入到Grade1表中。(2)

```
CREATE TABLE GRADE1
AS
SELECT *
FROM STUDENT S
WHERE SNO LIKE '99%' AND SNO IN (
    SELECT SNO
    FROM GRADE G
    WHERE CNO = (
        SELECT CNO
        FROM COURSE
        WHERE CNAME = '高等数学'
    )
);
```

69. 查询姓名中有“明”字的学生情况 (2)

```
SELECT *
FROM STUDENT
WHERE SNAME LIKE '%明%';
```

70. 查询姓名是三个字的学生情况 (2)

```
SELECT *
FROM STUDENT
WHERE LEN(SNAME) = 3;
```

71. 查询选修了3门以上课程的学生学号、姓名。 (2)

```sql
SELECT SNO, SNAME
FROM STUDENT S
WHERE 3 < (
    SELECT COUNT(*)
    FROM GRADE G
    WHERE S.SNO = G.SNO
);
```

72. 查询所有男学生及其宿舍住宿情况。 (2)

```sql
SELECT *
FROM STUDENT JOIN DORM
WHERE SEX = '男';
```

73. 查询所有课程的间接先修课情况。 (2)

```sql
SELECT C.CNO, C1.CPNO
FROM COURSE C JOIN COURSE C1 ON C.CPNO =
C1.CNO;
```

74. 查询所有学生及其选修课程情况。(2)

```sql
SELECT *
FROM STUDENT JOIN GRADE;
```

75. 将“计算机”学生的“信息系统”的分数置0分处理。 (2)

```
UPDATE GRADE
SET SCORE = 0
WHERE SNO IN (
    SELECT SNO
    FROM STUDENT
    WHERE DNO = (
        SELECT DNO
        FROM DEPARTMENT
        WHERE DNAME = '计算机'
    )
) AND CNO = (
    SELECT CNO
    FROM COURSE
    WHERE CNAME = '信息系统'
);
```

76. 在视图VIEW_1中找出名字中有“明”字的学生的学号、姓名、宿舍号。(2)

```
SELECT SNO, SNAME, DORMNO
FROM VIEW_1
WHERE SNAME LIKE '%明%';
```

77. 创建每门课的选修人数的视图，生成如下结果的视图：(2)

课程名　选修人数

XXX　XXX

```
SELECT CNAME AS 课程名，COUNT(SNO) AS 选修人数
FROM GRADE JOIN COURSE
GROUP BY CNO;
```

78. 删除“数学系”的相关信息。 (2)

```
DELETE FROM COURSE
WHERE DNO = (
    SELECT DNO
    FROM DEPARTMENT
    WHERE DNAME = '数学系'
);
DELETE FROM STUDENT
WHERE DNO = (
    SELECT DNO
    FROM DEPARTMENT
    WHERE DNAME = '数学系'
);
DELETE FROM GRADE
WHERE DNO = (
    SELECT DNO
    FROM DEPARTMENT
    WHERE DNAME = '数学系'
);
DELETE FROM DEPARTMENT
WHERE DNAME = '数学系';
```

79. 删除“2202”宿舍的相关信息。（宿舍撤消，住宿的学生重新待分配）(2)

```sql
UPDATE STUDENT
SET DORMNO = NULL
WHERE DORMNO = '2202';

DELETE FROM DORMNO
WHERE DORMNO = '2202';
```

80. 查询选修课程最多的学生的学号和姓名 (3)

```sql
SELECT TOP 1 SNO, SNAME
FROM STUDENT JOIN GRADE
GROUP BY SNO
ORDER BY COUNT(CNO) DESC;
```

81. 假设一个宿舍最多可以住5个人，为没有分配宿舍的同学给出他住宿的所有可能 (3)

```sql
SELECT SNO, SNAME, D.DORMNO
FROM (
    SELECT SNO, SNAME
    FROM STUDENT
    WHERE DORMNO IS NULL
), (
    SELECT DORMNO
    FROM DORM D1
    WHERE 5 > (
        SELECT COUNT(SNO)
        FROM STUDENT S1
        WHERE S1.DORMNO = D1.DORMNO
    )
) AS D;
```

82. 查询姓名中第二个字为“明”的学生情况 (3)

```
SELECT *
FROM STUDENT
WHERE SNAME LIKE '_明%';
```

83. 查询至少选修了‘1’号和‘2’号课程的学生号码。(3)

```
SELECT SNO
FROM STUDENT S
WHERE 2 = (
    SELECT COUNT(CNO)
    FROM GRADE G
    WHERE S.SNO = G.SNO AND G.CNO IN ('1','2')
);
```

84. 查询**至少**选修了‘990101’学生的全部课程的学生学号(3)。

```
SELECT S.SNO
FROM STUDENT S
WHERE NOT EXISTS (
    SELECT G1.CNO
    FROM GRADE G1
    WHERE G1.SNO = '990101' AND NOT EXISTS (
        SELECT *
        FROM GRADE G2
        WHERE G2.SNO = S.SNO AND G2.CNO =
G1.CNO
    )
);
```

85. 插入一条学生记录（960102，张丽，女，--，5，2303）。其中5系是“历史系”。(3)

```
INSERT INTO DEPARTMENT(DNO,DNAME)
VALUES('5','历史系');
INSERT INTO
STUDENT(SNO,SNAME,SEX,SAGE,DNO,DORMNO)
VALUES('960102', '张丽', '女', NULL, '5', '2303');
```

86. 将宿舍表中“2404”宿舍的修改为“3404”宿舍。(3)

```
UPDATE DORM
SET DORMNO = '3404'
WHERE DORMNO = '2404';
```

87. 删除“96”级所有学生的相关信息。(3)

```
DELETE FROM STUDENT
WHERE SNO LIKE '96%';
```

88. 创建user1、user2、user3用户，并将user1获得的STUDENT表的查询、插入权限，能够级联授权给user2，user2再将获得的STUDENT表的查询权限级联授权给user3。（以上操作，需要检查是否授权成功。） (3)

```
CREATE LOGIN user1 WITH PASSWORD = 'password';
CREATE USER user1 FOR LOGIN user1;

CREATE LOGIN user2 WITH PASSWORD = 'password';
CREATE USER user2 FOR LOGIN user2;

CREATE LOGIN user3 WITH PASSWORD = 'password';
CREATE USER user3 FOR LOGIN user3;

GRANT SELECT, INSERT ON STUDENT TO user1 WITH
GRANT OPTION;
GRANT SELECT, INSERT ON STUDENT TO user2 WITH
GRANT OPTION;
GRANT SELECT ON STUDENT TO user3;
```