1. Please create the six relation schemas using CREATE statement, and insert the data into the tables.
2. Table s：

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Data Type | specification | Meaning |
| sno | **char（8）** | Primary key | Student number |
| sname | char(10) | Not null | Student name |
| sex | char(1) | ‘m’ or ’f’ | gender |
| birthday | date |  | birthday |
| classno | Char(6) | Foreign key | Classnumber |
| Totalcredit | Smallint | Initial value is 0 | Total credit |

Create table s (sno char(8) primary key,

sname char(10) not null,

sex char(1),

birthday date,

classno char(6),

totalCredit Smallint);

INSERT INTO s VALUES ('08300010','s1','m','1991-10-01','rj0801',0);

INSERT INTO s VALUES ('08300012','s2','m','1990-8-8','Rj0801', 0);

INSERT INTO s VALUES ('08300015','s3','f','1990-5-22','Rj0801', 0);

INSERT INTO s VALUES ('08300020','s4','f','1989-1-8','Rj0802', 0);

INSERT INTO s VALUES ('08300030','s5','m','1990-10-8','Rj0802', 0);

INSERT INTO s VALUES ('08300048','s6','m','1989-6-6','Rj0803', 0);

INSERT INTO s VALUES ('08300050','s7','m','1990-6-10','Rj0803', 0);

INSERT INTO s VALUES ('08300067','s8','f','1990-8-21','Rj0803', 0);

INSERT INTO s VALUES ('08300075','s9','f','1991-9-23','Rj0804', 0);

INSERT INTO s VALUES ('08300088','s10','m','1989-3-1','Rj0804', 0);

INSERT INTO s VALUES ('08300100','s11','f','1990-2-26','Rj0805', 0);

INSERT INTO s VALUES ('08300148','s12','m','1991-4-25','Rj0805', 0);

INSERT INTO s VALUES ('08300150','s13','f','1989-11-15','Rj0805', 0);

INSERT INTO s VALUES ('08300160','s14','f','1990-12-12','Rj0806', 0);

INSERT INTO s VALUES ('08300161','s15','m','1991-10-12','Rj0806', 0);

INSERT INTO s VALUES ('08300160','s16','m','1990-11-6','Rj0806', 0);

INSERT INTO s VALUES ('08300160','s17','f','1990-12-22','Rj0806', 0);

1. Table class：

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Data Type | specification | Meaning |
| classno | char(6) | Primary key | Classnumber |
| className | char(20) | Not null | Classname |
| ClasssMajor | char(20) |  | major |
| school | char(40) |  | school |
| **studentNumber** | **smallint** | **[20..40]** | **The number of students in this class** |

Create table class (classno char(6) Primary key,

className char(20)Not null,

classsMajor char(20),

school char(40),

studentNumber smallint,

Check (studentNumber between 20 and 40));

INSERT INTO class VALUES ('Rj0801','Software0801','SoftwareEngineering','School of Software Engineering', 24);

INSERT INTO class VALUES ('Rj0802','Software0802','Software Engineering','School of Software Engineering', 26);

INSERT INTO class VALUES ('Rj0803','Software0803','SoftwareEngineering','School of Software Engineering', 25);

INSERT INTO class VALUES ('Rj0804','Software0804','SoftwareEngineering','School of Software Engineering', 25);

INSERT INTO class VALUES ('Rj0805','Software0805','SoftwareEngineering','School of Software Engineering', 24);

INSERT INTO class VALUES ('Rj0806','Software0806','SoftwareEngineering','School of Software Engineering', 24);

1. Table course：

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Data Type | specification | Meaning |
| cno | **char (6)** | Primary key | course number |
| cname | char(30) | Not null | Course name |
| credit | smallint | [1..4] | credit |

Create table course

( cno char(6) primary key,

cname char(30)Not null,

credit Smallint);

INSERT INTO course VALUES ('801','Foundation of Computer', 3);

INSERT INTO course VALUES ('802','Programming Language', 2);

INSERT INTO course VALUES ('803','Database Structure', 4);

INSERT INTO course VALUES ('804','Database System', 4);

INSERT INTO course VALUES ('805','Operating System', 3);

INSERT INTO course VALUES ('806','Computer Network', 4);

1. Table sc：

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Data Type | specification | Meaning |
| sno | char (8) | Foreign key | student number |
| cno | char (6) | Foreign key | Course number |
| grade | Smallint | [0..100] | score |

Create table sc(sno char(8),

cno char(6),

primary key(sno, cno),

grade Smallint

FOREIGN KEY(sno)REFERENCES s(sno)on delete cascade,

FOREIGN KEY(cno)REFERENCES course(cno)on delete cascade,

CHECK ((grade IS NULL) OR (grade BETWEEN 0 AND 100))

);

INSERT INTO sc VALUES ('08300010','801', 88);

INSERT INTO sc VALUES ('08300010','802', 55);

INSERT INTO sc VALUES ('08300010','803', 91);

INSERT INTO sc VALUES ('08300012','801', 88);

INSERT INTO sc VALUES ('08300012','802',null);

INSERT INTO sc VALUES ('08300012','803', 91);

INSERT INTO sc VALUES ('08300012','804', 78);

INSERT INTO sc VALUES ('08300015','801', 95);

INSERT INTO sc VALUES ('08300015','802', 67);

INSERT INTO sc VALUES ('08300015','804', 58);

INSERT INTO sc VALUES ('08300015','805', 89);

INSERT INTO sc VALUES ('08300015','806', 71);

INSERT INTO sc VALUES ('08300020','801', 95);

INSERT INTO sc VALUES ('08300020','803', 67);

INSERT INTO sc VALUES ('08300020','805', 58);

INSERT INTO sc VALUES ('08300020','806',null);

INSERT INTO sc VALUES ('08300030','801', 91);

INSERT INTO sc VALUES ('08300030','803', 74);

INSERT INTO sc VALUES ('08300030','805', 84);

INSERT INTO sc VALUES ('08300030','806', 56);

INSERT INTO sc VALUES ('08300048','801', 92);

INSERT INTO sc VALUES ('08300048','802', 62);

INSERT INTO sc VALUES ('08300048','803', 23);

INSERT INTO sc VALUES ('08300050','801', 32);

INSERT INTO sc VALUES ('08300050','802', 72);

INSERT INTO sc VALUES ('08300050','803', 82);

INSERT INTO sc VALUES ('08300050','804', 92);

INSERT INTO sc VALUES ('08300050','805', 42);

INSERT INTO sc VALUES ('08300067','802', 82);

INSERT INTO sc VALUES ('08300067','803', 76);

INSERT INTO sc VALUES ('08300067','804', 90);

INSERT INTO sc VALUES ('08300075','803', 79);

INSERT INTO sc VALUES ('08300075','806', 68);

1. Tableteacher：

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Data Type | specification | Meaning |
| tno | char(6) | Primary key | teacher number |
| tname | char(10) | Not null | teacher name |
| sex | char(1) | ‘m’ or ’f’ | gender |
| birthday | date |  | birthday |
| title | char(20) |  | title |

Create table teacher (tno char(6) primary key,

tname char(10) not null,

sex char(1),

birthday date,

titlechar(20) );

INSERT INTO teacher VALUES ('T01','t1','m','1980-6-10','lecturer');

INSERT INTO teacher VALUES ('T02','t2','f','1970-3-14','professor');

INSERT INTO teacher VALUES ('T03','t3','m','1973-4-20','associate professor');

INSERT INTO teacher VALUES ('T04','t4','m','1981-8-30','lecturer');

INSERT INTO teacher VALUES ('T05','t5','f','1975-7-20',' associate professor');

INSERT INTO teacher VALUES ('T06','t6','m','1980-9-19','lecturer');

1. Tableteaching：

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Data Type | specification | Meaning |
| tno | char(6) | Foreign key | Teacher number |
| cno | char(6) | Foreign key | Course number |
| language | char(10) | Chinese, Bilingual, English | Language in lecture |

Create table teaching

( tno char(6), cno char(6), Llanguagechar (10),

primary key(Tno, Cno),

FOREIGN KEY(tno)REFERENCES teacher(tno)on delete cascade,

FOREIGN KEY(cno)REFERENCES course(cno)on delete cascade

);

INSERT INTO teaching VALUES ('T01','801','English');

INSERT INTO teaching VALUES ('T02','801','Bilingual');

INSERT INTO teaching VALUES ('T03','802','English');

INSERT INTO teaching VALUES ('T04','802','Bilingual');

INSERT INTO teaching VALUES ('T05','803','English');

INSERT INTO teaching VALUES ('T06','803','Bilingual');

INSERT INTO teaching VALUES ('T01','804','English');

INSERT INTO teaching VALUES ('T03','804','Bilingual');

INSERT INTO teaching VALUES ('T03','805','English');

INSERT INTO teaching VALUES ('T04','805','Bilingual');

INSERT INTO teaching VALUES ('T03','806','English');

INSERT INTO teaching VALUES ('T06','806','Bilingual');

1. Simple SQL statements.
2. List the basic information of all students.
3. List sno, sname, birthday of all male students.
4. Add a column named ‘addr’in table S，then change its length from 20 to 25.
5. Add a column named ‘registerDate：date’ in table s，Set a default value for column registerDate as the current system date, then drop it.
6. Set a default value 18 for column age in table s.

1. Set column sname as alternate key (not primary key) in table s.
2. Create a unique index on ascending sno and descending cno in Table SC.
3. Add a constraint in table S :male students’ age should be younger than 23 and female students’ age should be younger than 21.
4. Create a view named View\_80 which stores the rows with scores more than 80 using ‘with check option’, list sno, cno, and grade.
5. List the tuples with scores more than 90.
6. Insert the following rows into View\_80.

08300010，801，87

08300010，804，76

1. In view View\_80, remove the following tuple : sno =08300010，cno=801

1. In view View\_80, update the following tuples:

Change the grade to 90 of tuple（08300010，803）

Change the grade to 70 of tuple（08300010，803）

1. List sno, sname, sex and birthday of female students who were born before 1980-01-01.
2. List sno, sname, sex and birthday of male students whose names contain ‘s1’.
3. Search for the teachers who give the lectures in English, displaying tno, name and number of courses.
4. List tno, name and title of teachers whose title is not lecturer.
5. List sno of students who took course(s) and did not take the exam(s).
6. List sno, grade of students who fail the exam, dispaying the tuples in descending order.
7. List tno, tname, birthday of teachers who were born in 1970.
8. List the number of students studying each course.
9. List the teacher numbers of teachers who teach more than 2 courses, showing the number of courses at the same time.
10. List the average score, minimum score and maximum score of course 801.
11. List the names, birthday of teachers whose titles are lecturers and who were born after 1960.
12. Complex SQL Statements
13. Create a view name new\_View, showing sno, sname, classno, cname and grade.
14. In new\_View, list sno, sname, cname and grade of the students in Class named Software0801.

1. In new\_View, insert the following value (08300168，s21，Rj0803，Database System，88)
2. List each student’s total credits, showing sno, sname and total credits.
3. List each student’s average score and the number of elective courses, showing sno, sname, average score and the number of elective courses.
4. Search for the students who took course(s) and did not take the exam(s) showing sno, sname, cno and cname.
5. Search for the students who took course(s) and did not pass the exam(s) showing sno, sname, cno , cname and grade.
6. Search for the students who take the course named “Programming Language”, showing sname and grade.
7. Search for the students in the class named “Software0801”, showing sno, sname, cno, cname and grade.
8. List the teaching information of all teachers, showing tname and cname.
9. List the information that the score is less than the average score of the same course, showing sno, sname, cname and **by how much**.
10. Search for the students who are in the same class with a student named ‘s1’.
11. Search for the students who do not take “Programming Language”, showing the student names.
12. Search for the teachers who teach both Data Structure and Database System, showing their names.
13. Search for the teachers who teach all courses, showing their names.
14. Find all female students who take both 801 and 802 as elective courses, list their names.
15. Find all students who takes neither 801 nor 802 as an elective course, list their student names.
16. Search for the female students who got a score of 95.
17. Search for the female students who take more than 3 courses.
18. Find the male students whose average grades are more than 80, list their student numbers and student names.
19. Search for the average score of each course which is taught by a teacher named ‘t1’.
20. Search for the number of male students of each age, list the age with the number of students more than 20, in descending order of the number of students.
21. Search for the students whose each grade is more than 90, showing their names.
22. Search for the male students who are older than all female students, showing their names.
23. Search for the female students who do not take course 802, showing the student names.
24. Search for the students who pass the exams of all courses which he takes, showing their names.
25. Find the students who take all the courses and list their names.
26. Search for the students who take all courses that are taken by ‘s1’, showing their names.
27. Search for the student whose average score is the highest.
28. Search for the students whose average score are higher than the average score of his or her class.

4．Exercise of DML

（1）Increase the grade by 5% for the female students who take course(s) taught by a teacher name ‘t1’

（2）**Find the students whose every grade is no less than 80. Insert the student numbers, names and sex into an existing table named STUD（sno，sname，sex）**

**（3）**In table sc, Remove the records in which there in no grade.

**（4）**Remove the score records about a student named ‘s1’

**（5）**Set the grade to null, which is less than 60 for course ‘Data Structure’.

**（6）**Increase the grade by 5% for the female students whose scores are less than the total average score.

**（7）**For course’804’ in table sc, increase the grade by 5% if grade<=75, increase the grade by 4% if grade between 76 and 95, using two update statements.

**5．**Define Stored Procedure

**（1）**Create a Stored Procedure named ‘insert\_s’, it can insert a record into Table s, including 5 parameters: sno, sname, sex, birthday and classno. And give the SQL statement how to call ‘insert\_s’ with parameter values.

**（2）**Create a Stored Procedure named ‘insert\_course’, it can insert a new course into Table course, including 3 parameters: cno, cname and credit, setting the default value 4 for credit. And give two cases of the SQL statement how to call ‘insert\_course’ with parameter values, one is providing the value of cno and cname, the other is providing cno, cname and credit. Please compare the results of two cases.

**（3）**Create a Stored Procedure named ‘query\_student’, it can search for a student information of sname, sex, birthday and classno according to a given sno. And give the SQL statement how to call ‘query\_student’ using sno “08300012”.

**（4）**Create a Stored Procedure named ‘select\_average’, it can input sname, sex, age and average score of all students whose average scores are more than 80. Please give the SQL statement how to call ‘select\_average’ , and give the result.

**（5）**Create a Stored Procedure named ‘select\_all’, it can display sno, sname, the number of electives, average score and total credits of the students who take the courses taken by a given student with parameter: student name. Please give the SQL statement how to call ‘select\_all’ with a given student name, and give the result.

**（6）**Create a Stored Procedure named ‘select\_less’, it can display sno, sname and average score of the students who average score are less than the average score of a given class with parameter: classno. Please give the SQL statement how to call ‘select\_less’ with a given classno, and give the result.

6．Define Triggers

（1）Create a Trigger named ‘display\_trigger’ on Table s, it can display all the students information when inserting a new student into Table s.

（2）Create a Trigger named ‘increase\_num’ on Table s, it can increase the number of students in the corresponding class by 1 when inserting a new student into that class in Table s.

（3）Create a Trigger named ‘decrease\_num’ on Table s, it can decrease the number of students in the corresponding class by 1 when deleting a student from that class in Table s.

（4）Create a Trigger named ‘transfer\_num’ on Table s, it can decrease by 1 the number of students in the original class and increase by 1 the number of students in the new class when transferring a student from the original class to a new class.

（5）Create a trigger named’ insert\_sc’, when inserting a non-existing sno value in Table s into Table sc, it can insert a new row with sno equal to the new value and set null value to other attributes except sno in Table s.

（6）Create a trigger named ‘No\_lower\_Grade\_Trig’ to stop the attempt to lower the grade of a student.

（7）Create a Trigger named ‘tri\_nottoomuch’ on Table sc, it can give warning 'insert or update rollback transaction' and reject the operation when the number of electives taken by a student exceeds 3.

（8）Create two Triggers named ‘**insert\_sc1** ‘ and ‘**delete\_sc**’ on Table sc, ‘**insert\_sc1’** can increase the total credits in Table s when a student takes a new course, and ‘**delete\_sc**’ can decrease the total credits in Table s when a student quit from a course.