数据库第二次作业

2. SELECT title FROM course WHERE dept\_name = "Comp.Sci." AND credits = 3;
3. SELECT DISTINCT student.ID FROM ((student NATURAL JOIN takes NATURAL JOIN section) JOIN (teaches NATURAL JOIN instructor) USING (course\_id,sec\_id,semester,year)) WHERE instructor.`name` = "Einstein";
4. SELECT MAX(salary) FROM instructor;
5. SELECT \* FROM instructor WHERE salary = (SELECT MAX(salary) FROM instructor);
6. SELECT course\_id,sec\_id,building,room\_no,time\_slot\_id,COUNT(ID) FROM student NATURAL JOIN takes NATURAL JOIN section WHERE semester = "Fall" AND `year` = "2009" GROUP BY course\_id,sec\_id;
7. SELECT MAX(enrollment) FROM (SELECT course\_id,sec\_id,building,room\_no,time\_slot\_id,COUNT(ID) AS enrollment FROM student NATURAL JOIN takes NATURAL JOIN section WHERE semester = "Fall" AND `year` = "2009" GROUP BY course\_id,sec\_id) AS yiyu;
8. WITH yiyu(course\_id,sec\_id,building,room\_no,time\_slot\_id,enrollment) AS (SELECT course\_id,sec\_id,building,room\_no,time\_slot\_id,COUNT(ID) AS enrollment FROM student NATURAL JOIN takes NATURAL JOIN section WHERE semester = "Fall" AND `year` = "2009" GROUP BY course\_id,sec\_id) SELECT course\_id,sec\_id FROM yiyu WHERE enrollment = (SELECT MAX(enrollment) FROM yiyu);
10. SELECT SUM(points \* credits) FROM student NATURAL JOIN takes NATURAL JOIN grade\_points NATURAL JOIN course WHERE student.ID = "12345";
11. SELECT SUM(points \* credits)/SUM(credits) FROM student NATURAL JOIN takes NATURAL JOIN grade\_points NATURAL JOIN course WHERE ID = "12345";
12. SELECT SUM(points \* credits)/SUM(credits) FROM student NATURAL JOIN takes NATURAL JOIN grade\_points NATURAL JOIN course GROUP BY ID;
14. UPDATE instructor SET salary = salary \* 1.1 WHERE dept\_name = "Comp.Sci.";
15. DELETE FROM course WHERE course\_id NOT IN (SELECT course\_id FROM section);
16. INSERT INTO instructor SELECT ID,`name`,dept\_name,10000 FROM student WHERE tot\_cred > 100;
18. SELECT COUNT(DISTINCT name) FROM accident NATURAL JOIN participate NATURAL JOIN person WHERE date BETWEEN "2009/01/01" AND "2009/12/31";
19. INSERT INTO accident VALUES(3222,"2019/3/22","fsdfss");

INSERT INTO participated SELECT o.driver\_id,c.license,3222,2232 FROM person AS p,owns AS o,car AS c WHERE p.name = "defxd" AND p.driver\_id = o.driver\_id AND o.license = c.license AND c.model = "dggg";

1. DELETE car WHERE model = "Mazda" AND license IN (SELECT license FROM person NATURAL JOIN owns WHERE name = "John Smith" );
3. SELECT ID,CASE score

WHEN score < 40 THEN

"F"

WHEN score < 60 THEN

"C"

WHEN score < 80 THEN

"B"

ELSE

"A"

END CASE FROM marks;

1. WITH yiyu(ID,score) AS (SELECT ID,CASE score

WHEN score < 40 THEN

"F"

WHEN score < 60 THEN

"C"

WHEN score < 80 THEN

"B"

ELSE

"A"

END CASE FROM marks)

SELECT score,COUNT(ID) FROM yiyu GROUP BY score;



SELECT dept\_name FROM department WHERE LOWER(dept\_name) LIKE "%sci%";



两者都不空的时候；两者都空或者本身p为空。

2. SELECT customer\_name FROM customer WHERE customer\_name NOT IN (SELECT customer\_name FROM borrower);
3. SELECT c.customer\_name FROM customer AS c JOIN customer AS o USING (customer\_street,customer\_city) WHERE c.customer\_name = "Smith";
4. SELECT DISTINCT customer\_name FROM depositor NATURAL JOIN account NATURAL JOIN customer WHERE customer\_city = "Harrison";
6. SELECT employee\_name,city FROM works NATURAL JOIN employee WHERE company\_name = "First Bank Corporation";
7. SELECT employee\_name,street,city FROM works NATURAL JOIN employee WHERE company\_name = "First Bank Corporation" AND salary > 10000;
8. SELECT employee\_name FROM works WHERE company\_name != "First Bank Corporation";
9. SELECT employee\_name FROM works WHERE salary > (SELECT MAX(salary) FROM works WHERE company\_name = "Small Bank Corporation");
10. WITH yiyu(city) AS (SELECT city FROM company WHERE company\_name = "Small Bank Corporation") SELECT company\_name FROM company WHERE city IN (SELECT city FROM yiyu);
11. WITH yiyu(num,company\_name) AS (SELECT COUNT(employee\_name) AS num,company\_name FROM works NATURAL JOIN company GROUP BY company\_name) SELECT company\_name FROM yiyu WHERE num = (SELECT MAX(num) SELECT FROM yiyu);
12. WITH yiyu(salary,company\_name) AS (SELECT AVG(salary) AS salary,company\_name FROM works NATURAL JOIN company GROUP BY company\_name) SELECT company\_name FROM yiyu WHERE salary > (SELECT salary FROM yiyu WHERE company\_name = "First Bank Corporation");
14. UPDATE employee set city = "Newtown" WHERE employee\_name = "Jones";
15. UPDATE works set salary = CASE salary

WHEN salary > 100000 THEN

salary \* 1.03

ELSE

salary \* 1.1

END CASE WHERE company\_name = "First Bank Corporation" AND employee\_name NOT IN (SELECT employee\_name FROM employee);

2. SELECT DISTINCT `name` FROM (student NATURAL JOIN takes) JOIN course USING (course\_id) WHERE student.dept\_name = "Comp.Sci.";
3. SELECT ID,`name` FROM student WHERE ID NOT IN (SELECT ID FROM student NATURAL JOIN takes WHERE `year` < 2009);
4. WITH yiyu(sa,dept) AS (SELECT MAX(salary) AS salary,`dept\_name` FROM instructor GROUP BY dept\_name) SELECT ID,`name`,dept\_name,salary FROM instructor WHERE (salary,dept\_name) IN (SELECT sa,dept FROM yiyu);
5. SELECT MIN(maxsalary) FROM (SELECT MAX(salary) AS maxsalary FROM instructor GROUP BY dept\_name) AS yiyu;
7. INSERT INTO course VALUES("CS-001","WeeklySeminar","Comp.Sci.",0);
8. INSERT INTO section VALUES("CS-001","1","Autumn",2009,NULL,NULL,NULL);
9. INSERT INTO takes SELECT ID,"CS-001","1","Autumn",2009,NULL FROM student WHERE dept\_name = "Comp.Sci.";
10. DELETE FROM takes WHERE course\_id = "CS-001" AND sec\_id = "1" AND `year` = 2019 AND semester = "Autumn" AND ID IN (SELECT ID FROM student WHERE `name` = "Chavez");
11. 直接删除它，据百度会阻止该操作，但是亲自实验，并不会阻止该操作，而是会将与之相关的元组全部删除掉。
12. DELETE FROM takes WHERE course\_id IN (SELECT course\_id FROM course WHERE LOWER(title) LIKE "%database%");

CREATE TABLE person

(

driver\_id VARCHAR(18),

name VARCHAR(18),

address VARCHAR(18),

PRIMARY KEY(report\_number)

);

CREATE TABLE car

(

license VARCHAR(18),

model VARCHAR(18),

year DECIMAL(4),

PRIMARY KEY(license)

);

CREATE TABLE accident

(

report\_number VARCHAR(18),

date VARCHAR(18),

location VARCHAR(18),

PRIMARY KEY(report\_number)

);

CREATE TABLE owns

(

driver\_id VARCHAR(18),

license VARCHAR(18),

PRIMARY KEY(driver\_id),

FOREIGN KEY(driver\_id) REFERENCES person(driver\_id),

FOREIGN KEY(license) REFERENCES car(license)

);

CREATE TABLE participate

(

report\_number VARCHAR(18),

license VARCHAR(18),

driver\_id VARCHAR(18),

damage\_amount VARCHAR(12),

PRIMARY KEY(report\_number,license),

FOREIGN KEY(report\_number) REFERENCES accident(report\_number),

FOREIGN KEY(driver\_id) REFERENCES person(driver\_id),

FOREIGN KEY(license) REFERENCES car(license)

);

2. SELECT count(report\_number) FROM participated NATURAL JOIN person WHERE `name` = "John Smith";
3. UPDATE participated SET amount = 3000 WHERE license="AABB2000" AND report\_number = "AR2197";
5. SELECT customer\_name FROM branch NATURAL JOIN account NATURAL JOIN depositor NATURAL JOIN customer WHERE branch\_city = "Brooklyn";
6. SELECT SUM(amount) FROM loan;
7. SELECT branch\_name FROM branch WHERE assets > SOME(SELECT assets FROM branch WHERE branch\_city = "Brooklyn");
9. SELECT employee\_name FROM works WHERE company\_name = "First Bank Corporation";
10. SELECT employee\_name FROM employee\_name NATURAL JOIN works JOIN company USING company\_name WHERE employee.city = company.city;
11. SELECT employee\_name FROM works WHERE city = SOME(SELECT city FROM employee NATURAL JOIN manages) AND employee\_name NOT IN (SELECT employee\_name FROM manages);
12. SELECT employee\_name FROM works AS w WHERE salary > (SELECT AVG(salary) FROM works AS o WHERE w.company\_name = o.company\_name);
13. WITH yiyu(company\_name,salary) AS (SELECT company\_name,SUM(salary) FROM works GROUP BY company\_name) SELECT company\_name FROM yiyu WHERE salary = (SELECT MIN(salary) FROM yiyu);
15. UPDATE works SET salary = salary \* 1.1 WHERE company\_name = "First Bank Corporation";
16. UPDATE works SET salary = salary \* 1.1 WHERE company\_name = "First Bank Corporation" AND employee\_name IN (SELECT employee\_name FROM manages);
17. DELETE FROM works WHERE company\_name = "Small Bank Corporation";

空值可以表示不确定的量，亦能表示出不存在的量，所以空值会被引入数据库。



<>是不等号的意思，那么<>all就表示不在所有的结果中，那么就和not in 是等价的了。

1. 不要求做。
3. SELECT name FROM member NATURAL JOIN borrowed NATURAL JOIN book WHERE publisher = "McGraw-Hill";
4. WITH yiyu(isb) AS (SELECT isbn,memb\_no FROM book WHERE publisher = "McGraw-Hill") SELECT name FROM member NATURAL JOIN borrowed WHERE NOT EXISTS (SELECT isb FROM yiyu NOT IN (SELECT isbn FROM member NATURAL JOIN borrowed));
5. WITH yiyu(name,m) AS (SELECT name,COUNT(memb\_no) FROM member NATURAL JOIN borrowed NATURAL JOIN book GROUP BY publisher) SELECT name FROM yiyu WHERE m > 5;
6. WITH yiyu(c) AS (SELECT COUNT(isbn) FROM borrowed GROUP BY memb\_no) SELECT AVG(c) FROM yiyu;
7. 不要求做

分组的四个属性是参照section表的外键，这使得takes中每个元组都能在section表中匹配到结果，同时又因为是takes表的主键，所以不为空。综上所述：from子句中连接section表不会导致数据改变，所以查询结果不会改变。



SELECT dept\_name FROM (SELECT dept\_name,SUM(salary) AS s,COUNT(dept\_name) AS c FROM instructor GROUP BY dept\_name) AS yiyu WHERE s >= (SELECT SUM(s) / c FROM (SELECT dept\_name,SUM(salary) AS s,COUNT(dept\_name) AS c FROM instructor) AS yiyu);