Homework 4

张海

3130000923

备注

作业 1、2、3 已以作业本形式提交。

4.7

```
 CREATE TABLE employee (
        employee_name VARCHAR(20) NOT NULL,
        street VARCHAR(20) NOT NULL,
        city VARCHAR(20) NOT NULL,
        PRIMARY KEY (employee_name)
);

CREATE TABLE company (
        company_name VARCHAR(20) NOT NULL,
        city VARCHAR(20) NOT NULL,
        PRIMARY KEY (company_name)
);

CREATE TABLE works (
        employee_name VARCHAR(20) NOT NULL,
        company_name VARCHAR(20) NOT NULL,
        salary VARCHAR(20) NOT NULL,
        PRIMARY KEY (employee_name),
        FOREIGN KEY (employee_name) REFERENCES employee ON DELETE CASCADE,
        FOREIGN KEY (company_name) REFERENCES com
pany ON DELETE CASCADE
);

CREATE TABLE manages (
        employee_name VARCHAR(20) NOT NULL,
        manager_name VARCHAR(20) NOT NULL,
        PRIMARY KEY (employee_name),
        FOREIGN KEY (employee_name) REFERENCES em
ployee ON DELETE CASCADE,
        FOREIGN KEY (manager_name) REFERENCES emp
loyee ON DELETE CASCADE
)
```

4.12

With outer join:

```
 SELECT employee_name FROM employee NATURAL LEFT
OUTER JOIN manages WHERE manager_name IS NULL);
```

Without outer join:

```
 (SELECT employee_name FROM employee) EXCEPT (SEL
ECT manages.employee_name FROM employee NATURAL J
OIN manages WHERE manager_name IS NOT NULL);
```

5.15

a.

```
 CREATE FUNCTION average_salary (cname varchar(20
)) RETURNS INT BEGIN
    RETURN (SELECT AVG(salary) FROM works WHERE c
ompany_name == cname);
END
SELECT company_name FROM works GROUP BY company_name HAVING AVG(salary) > average_salary("First Bank Corporation");
```

b.

```
 SELECT company_name FROM works GROUP BY company_name HAVING AVG(salary) > (SELECT AVG(salary) FROM works WHERE company_name == "First Bank Corporation");
```

5.17

Embedded SQL is useful when imperative actions like displaying results and interacting with the user are needed.

SQL functions are primarily a mechanism for extending the power of SQL to handle attributes of complex data types (like images), or to perform complex and non-standard operations.

5.21

```
 CREATE TRIGGER r_B_s_A_on_delete_cascade AFTER DELETE OF A ON s
        REFERENCING old_row AS orow
        FOR EACH ROW BEGIN ATOMIC
    DELETE FROM r WHERE B = orow.A
        END
```