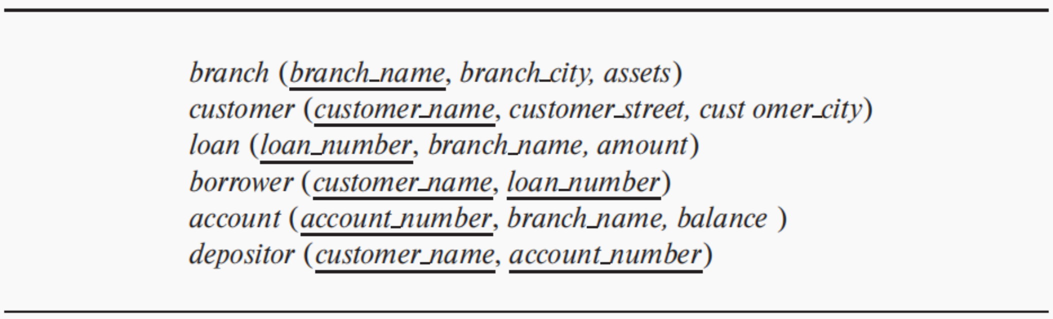
数据库作业week5

5.6

> Consider the bank database of Figure 5.21. Let us define a view *branch cust* as follows:
>
> ```
> create view branch_cust as
> select branch name, customer name
> from depositor, account
> where depositor.account number = account.account number
> ```
>
> Suppose that the view is *materialized*; that is, the view is computed and stored. Write triggers to *maintain* the view, that is, to keep it up-to-date on insertions to *depositor* or *account*. It is not necessary to handle deletions or updates. Note that, for simplicity, we have not required the elimination of duplicates.



branch (*branch_name*, *branch_city*, *assets*)
customer (*customer_name*, *customer_street*, *cust omer_city*)
loan (*loan_number*, *branch_name*, *amount*)
borrower (*customer_name*, *loan_number*)
account (*account_number*, *branch_name*, *balance*)
depositor (*customer_name*, *account_number*)

Figure 5.21 Banking database for Exercise 5.6.

```
create trigger update_bank1
after insert on depositor
referencing new row as new
for each row
insert into branch_cust
select branch_name,new.customer_name
from account
where account.account_number = new.account_number;

create trigger update_bank2
after insert on account
referencing new row as new
for each row
insert into branch_cust
select new.branch_name,customer_name
from depositor
where new.account_number = depositor.account_number;
```

5.15

> Consider an employee database with two relations

> *employee* (*employee_name*, *street*, *city*)
> *works* (*employee_name*, *company_name*, *salary*)
>
> where the primary keys are underlined. Write a function *avg salary* that takes a company name as an argument and finds the average salary of employees at that company. Then, write an SQL statement, using that function, to find companies whose employees earn a higher salary, on average, than the average salary at “First Bank”.

```
create function avg_salary (company_name varcahr(20))
returns numeric(10,2)
begin
    declare avg_salary numeric(10,2);
    select avg(salary) into avg_salary
    from works
    where works.company_name = company_name;
    return avg_salary;
end

select distinct company_name
from works
where avg_salary(works.company_name)>avg_salary('First Bank');
```

5.19

> Suppose there are two relations *r* and *s*, such that the foreign key *B* of *r* references the primary key *A* of *s*. Describe how the trigger mechanism can be used to implement the **on delete cascade** option when a tuple is deleted from *s*.

```
create trigger update_r before delete on s
referencing old row as olds
for each row
begin
    delete
    from r
    where r.B = olds.A;
end

## 如上面的trigger，删除s的元组前，触发器将所有外键指向该元组值的元组从r中删掉，然后删掉s的该元组。
## As in the trigger above, before deleting the tuple of S, the trigger deletes all tuples with foreign keys pointing to the tuple value from R, and then deletes the tuple of S.
```