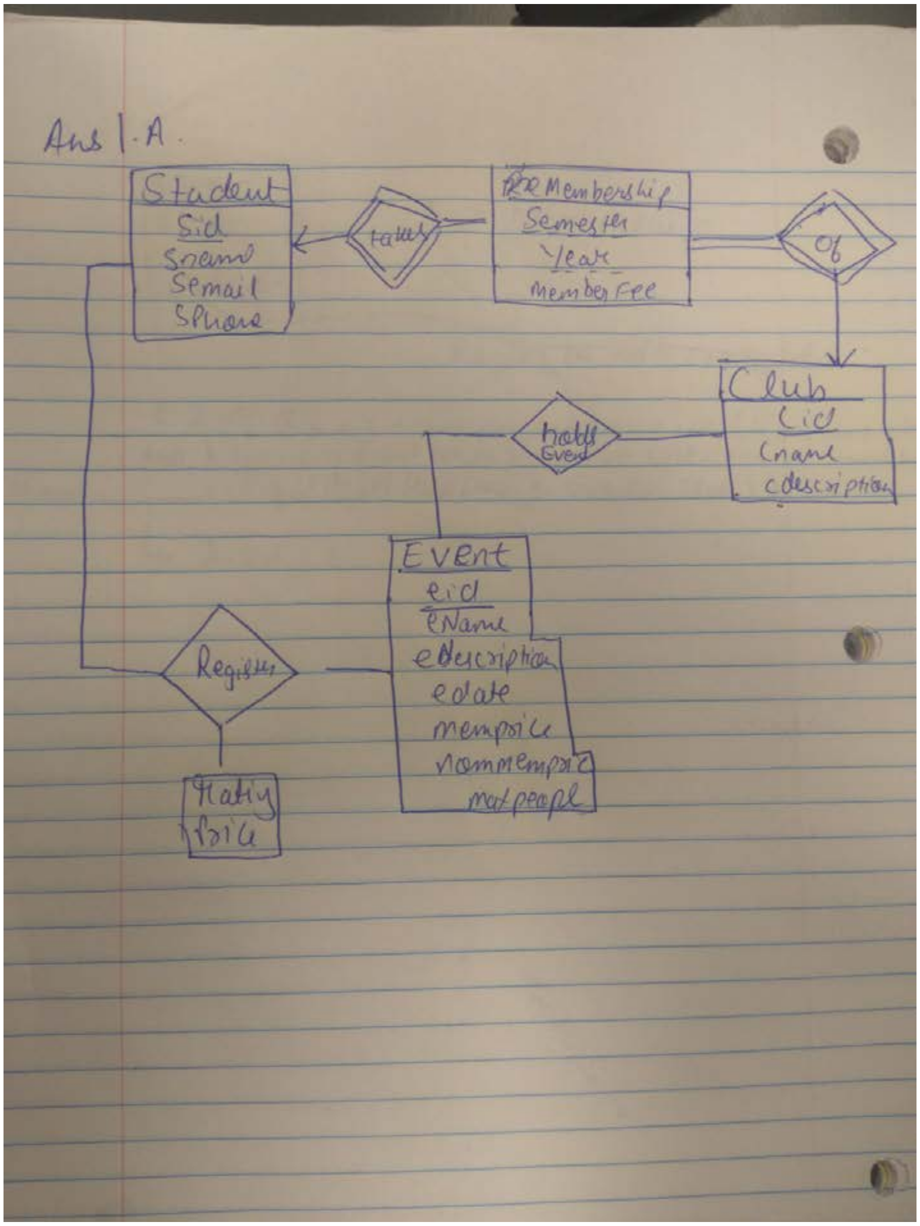
NYU School of Engineering **Oct 29th, 2018**
Computer Science and Engineering
CS 6083, Fall 2018

Problem Set #2 Sample Solution

Problem 1.

(a)



Student (sid, sname, semail, sphone);
Club (cid, cname, cdescription);
Event (eid, ename, edescription, edate, memprice, nonmemprice, maxpeople)
Membership (sid, cid, semester, year, memberfee);
HoldsEvent (eid, cid);
Register(eid, sid, price, rating);

(b)
Foreign keys → Referenced Table
Membership(sid) → Student(sid)
Membership(cid) → Club(cid)
HoldsEvent(eid) → Event(eid)
HoldsEvent(cid) → Club(cid)
Register(eid) → Event(eid)
Register(sid) → Student(sid)

(c)
Remove the HoldsEvent table and put cid in Event table.

Event (eid, cid, ename, edescription, edate, memprice, nonmemprice, maxpeople)

(d)
Remove memberfee from Event and put it in HoldsEvent.
HoldsEvent (eid, cid, price);

(e)

```
drop schema dbhw2;
create schema dbhw2;
use dbhw2;
create table Student (
    sid integer auto_increment primary key,
    sname varchar(50),
            semail varchar(50),
            sphone varchar(50));



create table Club(
         cid integer auto_increment primary key,
         cname varchar(50),
            cdescription varchar(50));
create table Event(
         eid int auto_increment primary key,
            ename varchar(50),
            edescription varchar(50),
            edate datetime,
            memprice int,
            nonmemprice int,
            maxpeople int
                ); 

create table Membership(
        sid int,
              cid int,
              semester varchar(50),
              year int,
              memberfee int,
              primary key(sid,cid,semester,year),
              FOREIGN KEY (sid) REFERENCES Student(sid) ON DELETE CASCADE,
              FOREIGN KEY (cid) REFERENCES Club(cid) ON DELETE CASCADE
                        );

create table HoldsEvent(
        eid int,
              cid int,
              FOREIGN KEY (eid) REFERENCES Event(eid) ON DELETE CASCADE,
              FOREIGN KEY (cid) REFERENCES Club(cid) ON DELETE CASCADE
);

create table Register(
        eid int,
              sid int,
              price int,
              rating float,
              FOREIGN KEY (eid) REFERENCES Event(eid) ON DELETE CASCADE,
              FOREIGN KEY (sid) REFERENCES Student(sid) ON DELETE CASCADE

);
```

(f)

(i)

```
select s.sid,ifnull(numberofevents,0) as 'number of events', totalprice from (
select sid, sum(price) as 'totalprice', count(eid) as 'numberofevents' from Register natural join
Event where edate > '2018-09-1 00:00:00'
and edate < '2019-01-1 00:00:00' group by sid) as t1 right outer join Student s on t1.sid=s.sid

-- using the rightouter join to get all the students that did not register for any event
```

(ii)

```
select eid, ename from HoldsEvent natural join Event group by eid having count(cid) = (select
max(counts) from
(select eid,count(cid) as counts from HoldsEvent group by eid) as  t2)
```

(iii)

```
create temporary table t1
select r.eid,r.sid,price,memprice,c.cid from Register r natural join HoldsEvent h, Event e, Club c
where h.eid = e.eid and h.cid=c.cid and  edate > '2017-09-1 00:00:00'
and edate < '2018-01-1 00:00:00' and cname='chess club' and (r.sid,h.cid) not in (select
mr.sid,mr.cid from Membership mr)  ;

select sid from t1
group by cid,sid having sum(price) > sum(memprice)+(select memberfee from Membership
where cid in (select cid from Club where cname = 'chess club') limit 1);
```

(iv)

```
select sid,cid from Membership where year in (2016,2017,2018) or
 (semester = 'spring' and year = 2015) group by sid,cid having count(*) = 7
```

(v)

```
create temporary table t1
select eid , avg(rating) as avg from Register group by eid;
create temporary table t2
select cid, min(avg) minavg from t1 natural join HoldsEvent group by cid;
create temporary table t3
select cid from t2 where minavg >=4;


-- get all the clubs that have not received any rating ( either because they didn't
-- held and event or no one attended it
create temporary table t4
select cid from Club where cid not in (Select cid from HoldsEvent natural join Register);

select cid from t3 union select * from t4;
```

(vi)

```
select he1.cid,he2.cid, count(*) count from HoldsEvent he1, HoldsEvent he2 where he1.eid =
he2.eid and he1.cid>he2.cid group by he1.cid , he2.cid
having count >= 5
```

(vii)

Option 1

```
create temporary table t1
select h1.cid as cid1,h2.cid as cid2, h1.eid from HoldsEvent h1,HoldsEvent h2 where h1.eid =
h2.eid and
h1.cid>h2.cid;
create temporary table t2
select cid from Club where cname = 'Japanese student association';
create temporary table t3
select cid from Club where cname = 'chinese student association';


select sname from Membership m natural join Student natural join Register r,t2,t3, t1
where t1.eid = r.eid and
(t1.cid1 = t2.cid and t1.cid2 = t3.cid)  or
(t1.cid1 = t3.cid and t1.cid2 = t2.cid)  and m.cid not in (select cid from club where cname =
'Japanese student association' or cname = 'chinese student association')
```

option 2

```
select sname from Student where sid in (

select sid from HoldsEvent he natural join Club c, Register r where r.eid=he.eid group by

sid having group_concat(cname ORDER BY cname DESC SEPARATOR ',') = 'Japanese
student association,chinese student association'

)
```

(g)

iii

$t_0 \leftarrow \Pi_{mr.sid,\ mr.cid}(\text{Membership as } mr)$

$T_1 \leftarrow \Pi_{r.eid,\ r.sid,\ price,\ memprice,\ c.cid}(\sigma_{h.eid=e.eid \land h.cid=c.cid \land edate>'2017\text{-}09\text{-}1\ 00{:}00{:}00' \land edate<'2018\text{-}01\text{-}1\ 00{:}00{:}00' \land cname='chess\ club'}(\text{Registor as } r \bowtie \text{HoldsEvent as } h \times \text{Event as } e \times \text{Club as } c))$

$T_3 \leftarrow T_1 - T_0$

$T_4 \leftarrow \Pi_{sid}(\sigma_{sum(price) > sum(memprice)}({}_{cid,\ sid}G_{sum(price),\ sum(memprice)}(t_3))).$

$T_5 \leftarrow \Pi_{cid}(\sigma_{cname='chess'}(\text{Club}))$

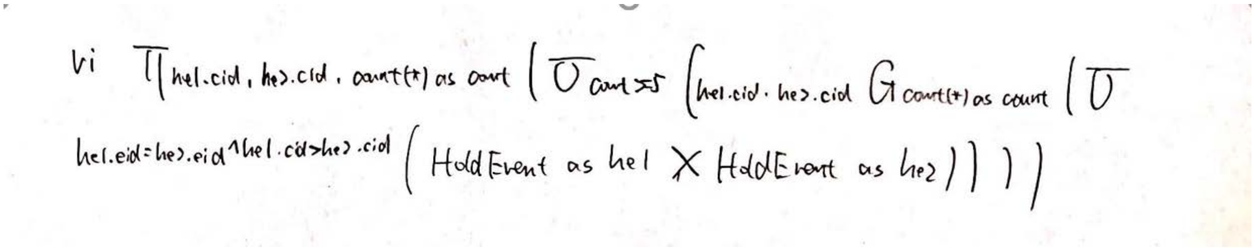
$T_6 \leftarrow \Pi_{memberfee}(\sigma_{cid\ in\ T_5 \land limit1}(\text{Membership}))$ $T_4 \cup T_6$

iv. $\Pi_{sid,\ cid}(\sigma_{count(*)=7}({}_{sid,\ cid}G_{count(*)}(\sigma_{year\ in(2016,2017,2018) \lor (semester='spring'\ and\ year=2015)}(\text{Membership}))))$

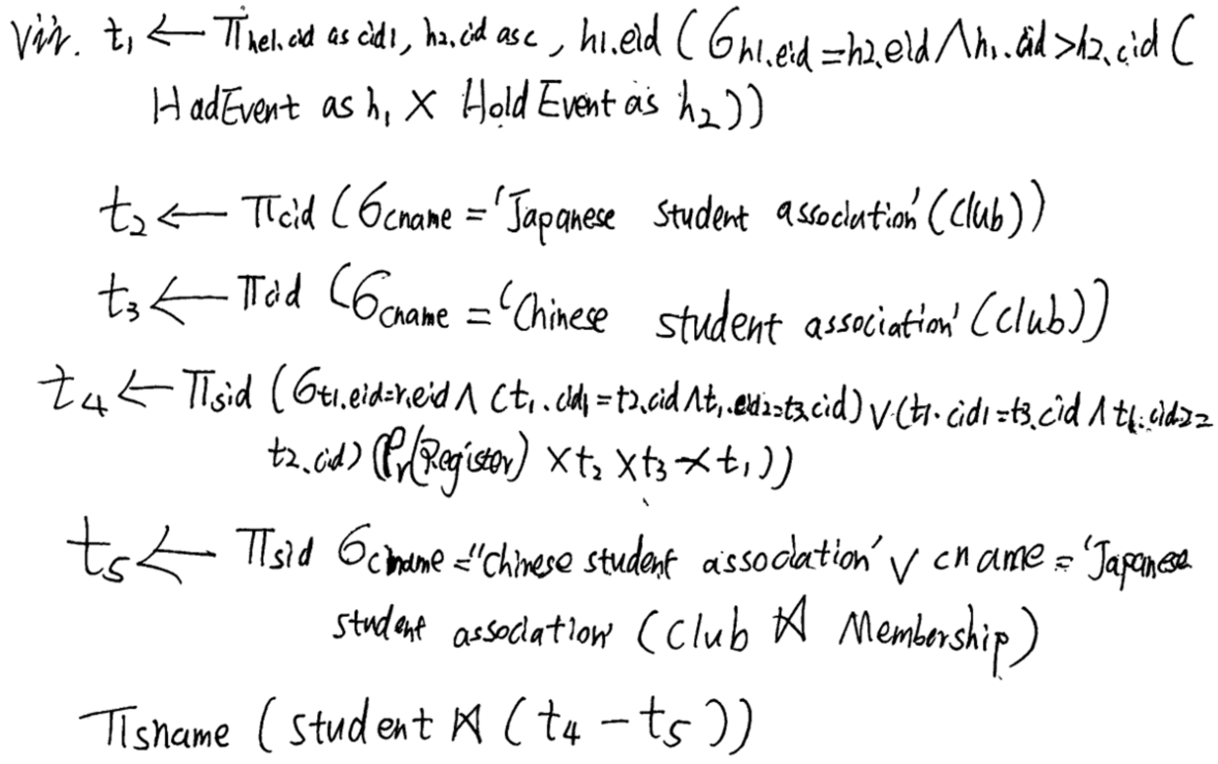
v $t_1 \leftarrow \Pi_{eid,\ avg}({}_{eid}G_{avg(rating)\ as\ avg}(\text{Registor}))$

$t_2 \leftarrow \Pi_{cid,\ minavg}({}_{cid}G_{min(avg)\ as\ minavg}(t_1 \bowtie \text{HoldsEvent}))$

$\Pi_{cid}(\sigma_{minavg>4}(t_2))$



vi $\Pi_{he1.cid,\ he2.cid,\ count(*)\ as\ count}(\sigma_{count>=5}({}_{he1.cid \cdot he2.cid}G_{count(*)\ as\ count}(\sigma_{he1.eid=he2.eid \wedge he1.cid>he2.cid}(HoldEvent\ as\ he1 \times HoldEvent\ as\ he2))))$



viii. $t_1 \leftarrow \Pi_{he1.cid\ as\ cid1,\ h2.cid\ asc,\ h1.eid}(\sigma_{h1.eid=h2.eid \wedge h_1.cid>h2.cid}(HadEvent\ as\ h_1 \times HoldEvent\ as\ h_2))$

$t_2 \leftarrow \Pi_{cid}(\sigma_{cname='Japanese\ student\ assoclution'}(Club))$

$t_3 \leftarrow \Pi_{cid}(\sigma_{cname='Chinese\ student\ association'}(club))$

$t_4 \leftarrow \Pi_{sid}(\sigma_{t1.eid=r.eid \wedge (t_1.cid_1=t2.cid \wedge t_1.cid2=t3.cid) \vee (t1.cid1=t3.cid \wedge t_1.cid2=t2.cid)}(\rho_r(Registor) \times t_2 \times t_3 \times t_1))$

$t_5 \leftarrow \Pi_{sid}\ \sigma_{cname='Chinese\ student\ association' \vee cname='Japanese\ student\ association'}(Club \bowtie Membership)$

$\Pi_{sname}(student \bowtie (t_4 - t_5))$

(h)

(i)

insert into Membership (sid,cid,semester,year,memberfee) (

SELECT sid,cid,'Fall',2018, 5 from Register r natural join Event e, HoldsEvent he

where he.eid=e.eid and (sid,cid) not in (select sid,cid from Membership) and

edate > '2018-09-1 00:00:00'

and edate < '2019-01-1 00:00:00' group by sid,cid having count(*) > 5

);

(ii)

```
SET SQL_SAFE_UPDATES=0;

delete from Student where sid not in (select r.sid from Register r union select m.sid from Membership m) ;

SET SQL_SAFE_UPDATES=1;
```

(iii)

```
insert into Event (ename, edescription, edate, memprice,nonmemprice,maxpeople) values
        ("Halloween Party", "Halloween Party description", now(), 5,8, 50);


-- eid is auto increment, see the eid and cid from the tables
insert into HoldsEvent(eid,cid) values
                                 (21,17);
insert into HoldsEvent(eid,cid) values
                                 (21,16);
```

Problem 2

(a)

```
create view stuMem as

select sid, sname, cid, cname

from Student natural left outer join Membership natural left  join Club

where semester="Fall" and year=2018;
```

(i)

```
select count(cid) from stuMem

where sname="John Myers";
```

(ii)

```
create view stuEvent as

select sid, sname, count(eid) as event_num

from stuMem  natural join HoldsEvent natural join Event natural join Register

where edate between "2018-09-01" and "2018-12-31"

 group by sid, sname
```

(iii) For student(s) who joined more than 3 events as a member in Fall 2018, output sname of the student(s) and ename of event they attend

```
select sname, ename from stuEvent natural join Event
where event_num >= 3
```

(iv) drop view stuEvent, stuMem;

(b)

Can’t write a trigger. But an event which runs every midnight can achieve similar effect. Here is the example:

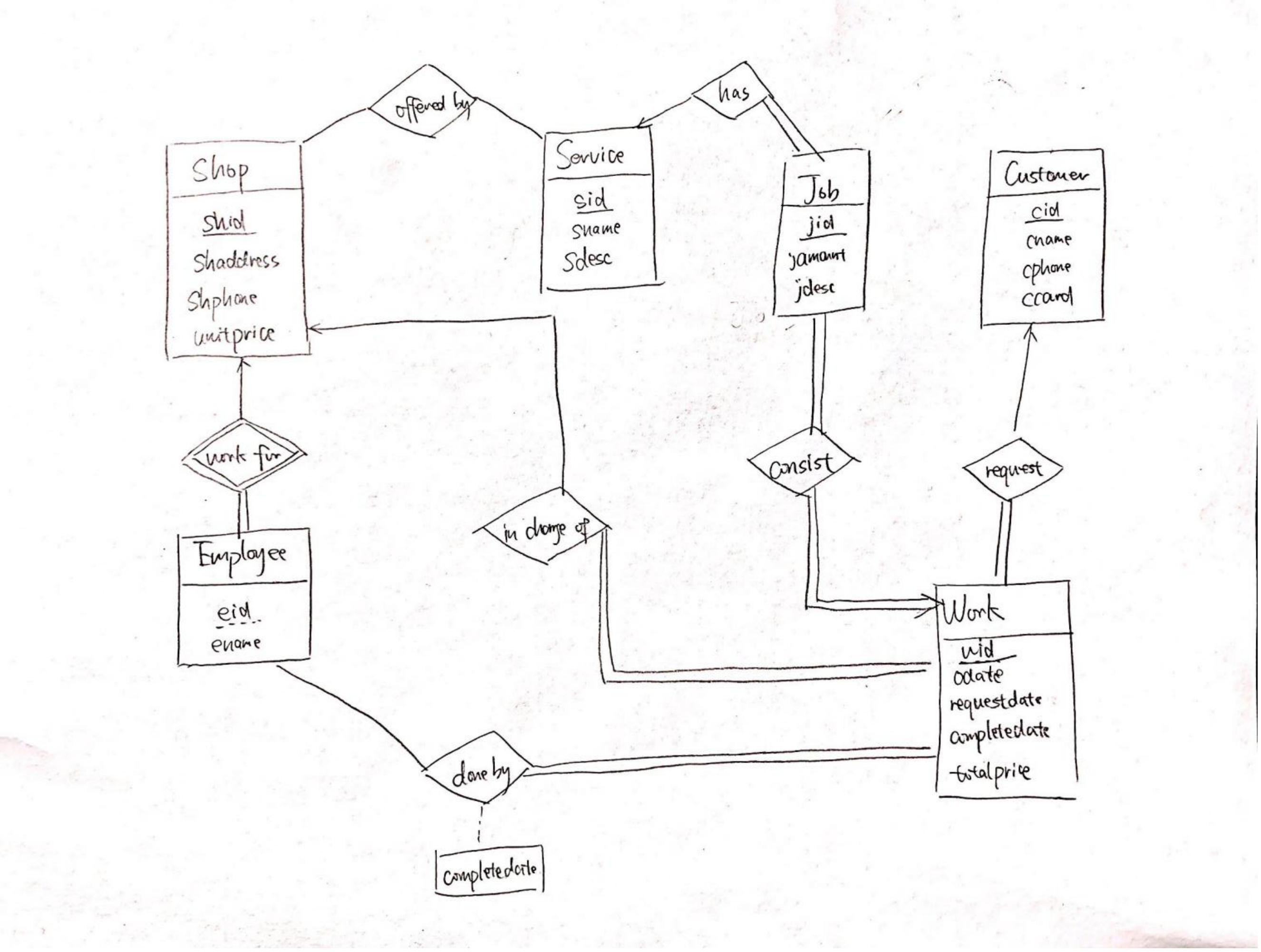
```
CREATE EVENT my_event
  ON SCHEDULE
    EVERY 1 DAY
    STARTS '2018-10-30 00:20:00' ON COMPLETION PRESERVE ENABLE
  DO
    delete from Student
    where sid not in (select sid from register natural join Event
    where datediff(now(), edate)<365*2);
```

(c)

```
CREATE trigger hw2c
After INSERT on Register
FOR EACH ROW
IF
(SELECT count (*)
FROM Register
WHERE NEW.eid = eid ) >= (SELECT 0.8 * maxpeople FROM EVENT WHERE NEW.eid = eid)
AND (SELECT count (*) -1
FROM Register
WHERE NEW.eid = eid ) < (SELECT 0.8 * maxpeople FROM EVENT WHERE NEW.eid = eid)
THEN
UPDATE EVENT
SET nonmemprice = 1.5*nonmemprice
WHERE NEW.eid = eid;
END IF;
```

Problem 3

(a)



(b)

Assumption here:

If you assume eid is global unique, then Employee is strong entity here. And Employee should be (eid, shid, ename). E.g If you assume each shop give the # just from 1 to 100, then the employee will be weak entity and the result will be shown as below.

Shop (<u>shid</u>, shaddress, shphone, unitprice)

Service (sid, sname, sdesc)

Offeredby (shid, sid)

Employee (eid, shid, ename)

Customer (cid, cname, cphone, ccard)

Work (wid, cid, shid, odate, requestdate, completedate, totalprice)

Job (jid, sid, wid, jamount, jdesc)

Doneby (wid, eid, completedate)

Offeredby.shid references Shop.shid

Offeredby.sid references Service.sid

Employee.shid references Shop.shid

Word.cid references Customer.cid

Word.shid references Shop.shid

Job.sid references Service.sid

Doneby.wid references Work.wid

Doneby.eid references Employee.eid

(c)

```
  (i) SELECT COUNT(wid), SUM(totalprice)
      FROM Work NATURAL JOIN Customer
      WHERE cname = "John Myers" AND YEAR(odate) = 2017
       GROUP BY cid;


   (ii) SELECT shid
       FROM Offeredby
       GROUP BY shid
       HAVING COUNT(DISTINCT sid) = (
                    SELECT COUNT(DISTINCT sid)
                    FROM Service);
```

```
    (iii) CREATE temporary table T3 as(
                SELECT T2.eid,
                CONCAT(ROUND((T1.cnt_late/T2.cnt_all * 100), 2), '%') AS percentage
                FROM (
                SELECT eid, COUNT(wid) AS cnt_late
                 FROM Work NATURAL JOIN Doneby
                WHERE DAY(completedate) - DAY(requestdate) >= 1
                GROUP BY eid) AS T1
         RIGHT JOIN (
                SELECT eid, COUNT(wid) AS cnt_all
                FROM Work NATURAL JOIN Doneby
                GROUP BY eid) AS T2
          ON T1.eid = T2.eid);
         UPDATE T3 set T3.percentage = 0 where T3.percentage is null;
         SELECT * from T3;
```

```
   (iv)   SELECT DISTINCT cid, cname
        FROM Customer NATURAL JOIN Work NATURAL JOIN
                 (SELECT wid, jid, sname FROM Consist NATURAL JOIN Job
        NATURAL JOIN Service) AS R1,
                (SELECT wid, jid, sname  FROM Consist  NATURAL JOIN Job
        NATURAL JOIN Service) AS R2
        WHERE R1.wid = R2.wid AND R1.jid != R2.jid AND
                 R1.sname = "color copying" AND R2.sname = "shipping";
```

```
(d)  CREATE TABLE Shop (shid integer auto_increment primary key,
          shaddress varchar(50),
          shphone varchar(50),
         unitprice int);
```

```
CREATE TABLE Service (sid integer auto_increment primary key,
                sname varchar(50),
                sdesc varchar(50));

CREATE TABLE Offeredby (shid int, sid int);

CREATE TABLE Employee (eid integer auto_increment primary key,
                shid int,
                ename varchar(50));


CREATE TABLE Customer (cid integer auto_increment primary key,
                cname varchar(50),
                cphone varchar(50),
                ccard varchar(20));

CREATE TABLE Work (wid integer auto_increment primary key,
                cid int,
                shid int,
                odate datetime,
                requestdate datetime,
                completedate datetime,
                totalprice  int);

CREATE TABLE Job (jid integer auto_increment primary key,
             sid int,
             jamount int,
             jdesc varchar(50));
```

```
CREATE TABLE Consist (wid int, jid int);

CREATE TABLE Doneby (wid int,eid int, completedate
datetime);

INSERT INTO Shop (shaddress, shphone, unitprice) VALUES
("add1", "shop1111", 1);
INSERT INTO Shop (shaddress, shphone, unitprice) VALUES
("add2", "shop2222", 2);
INSERT INTO Shop (shaddress, shphone, unitprice) VALUES
("add3", "shop3333", 3);
INSERT INTO Shop (shaddress, shphone, unitprice) VALUES
("add4", "shop4444", 4);
INSERT INTO Shop (shaddress, shphone, unitprice) VALUES
("add5", "shop5555", 5);

INSERT INTO Service (sname, sdesc) VALUES
                ("printing", "print"),
                ("scaning", "scan"),
                ("color copying", "color copy"),
                ("binding", "bind"),
                ("shipping", "ship");

INSERT INTO Offeredby (shid, sid) VALUES
                      (1, 1),
                      (1, 2),
                      (1, 3),
                      (1, 4),
                      (1, 5),
                (2, 3),
                (2, 1),
                (3, 4),
                (4, 5),
                (5, 3),
                (5, 5);

INSERT INTO Employee (shid, ename) VALUES
            (1, "emp1"),
            (1, "emp2"),
            (1, "emp3"),
            (1, "emp4"),
            (1, "emp5"),
            (1, "emp6"),
            (1, "emp7"),
            (2, "emp8"),
            (3, "emp9"),
            (4, "emp10"),
            (5, "emp11"),
            (5, "emp12");

INSERT INTO Customer (cname, cphone, ccard) VALUES
            ("Mike", "cust1111", "credit1111"),
            ("John Myers", "cust2222", "credit2222"),
            ("Mary", "cust3333", "credit3333"),
            ("Tom", "cust4444", "credit4444"),
            ("Lily", "cust5555", "credit5555");

INSERT INTO Work (cid, shid, odate, requestdate, completedate, totalprice) VALUES
    (2, 1, "2017-10-09 05:35:35", "2017-10-11 05:35:35", "2017-10-12 05:35:35", 6),
    (2, 2, "2017-10-09 05:35:35", "2017-10-11 05:35:35", "2017-10-12 05:35:35", 9),
    (2, 3, "2017-10-09 05:35:35", "2017-10-11 05:35:35", "2017-10-12 05:35:35", 8),
    (2, 4, "2017-10-09 05:35:35", "2017-10-11 05:35:35", "2017-10-10 05:35:35", 7),
    (2, 5, "2017-10-09 05:35:35", "2017-10-11 05:35:35", "2017-10-12 05:35:35", 10),
    (2, 5, "2018-10-09 05:35:35", "2018-10-11 05:35:35", "2018-10-12 05:35:35", 5),
    (4, 1, "2017-10-09 05:35:35", "2017-10-11 05:35:35", "2017-10-12 05:35:35", 3),
    (3, 1, "2017-10-09 05:35:35", "2017-10-11 05:35:35", "2017-10-10 05:35:35", 2),
    (5, 1, "2018-10-09 05:35:35", "2018-10-11 05:35:35", "2018-10-10 05:35:35", 1);

INSERT INTO Job (sid, jamount,jdesc) VALUES
        (3, 10, "NA"),
        (3, 7, "NA"),
        (3, 6, "NA"),
        (1, 5, "NA"),
        (2, 4, "NA"),
        (4, 3, "NA"),
        (5, 2, "NA"),
        (5, 1, "NA");



INSERT INTO Consist (wid,jid) VALUES
            (1, 1),
            (1, 2),
            (3, 4),
            (4, 5),
            (5, 7),
            (5, 8),
            (1, 1),
            (1, 7),
                    (2, 8);

INSERT INTO Doneby (wid, eid, completedate) VALUES
                        (1, 2, "2017-10-12 05:35:35"),
                        (7, 2, "2017-10-12 05:35:35"),
                        (8, 2, "2017-10-10 05:35:35"),
                        (9, 2, "2018-10-10 05:35:35"),
                        (2, 8, "2017-10-12 05:35:35"),
                        (3, 9, "2017-10-12 05:35:35"),
                        (4, 10, "2017-10-10 05:35:35"),
                        (5, 11, "2017-10-12 05:35:35"),
                        (5, 11, "2017-10-12 05:35:35");
```