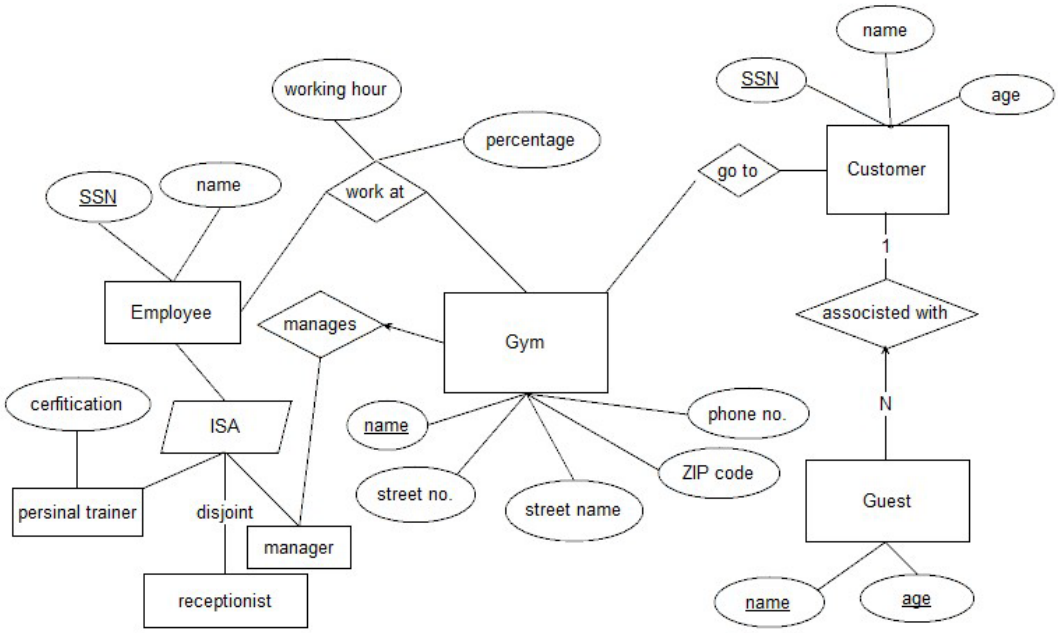
Homework 1

Haocong Wang

mw814@scarletmail.rutgers.edu

Question 1

(1)



(2)

```
CREATE TABLE gym
(
   name CHAR(20) NOT NULL PRIMARY KEY,
   streer_no INT,
   street_name CHAR(40),
   zip_code INT,
   manager CHAR(11) NOT NULL,
   FOREIGN KEY (manager) REFERENCES employ (SSN)
);

CREATE TABLE employee
(
   SSN CHAR(11) NOT NULL PRIMARY KEY,
   name CHAR(20)
   specialization CHAR(20)
);
```

```
CREATE TABLE customer
(
   SSN CHAR(11) NOT NULL PRIMARY KEY,
   name CHAR(20),
   age INT
);

CREATE TABLE phone_no
(
   phone_no LONG NOT NULL PRIMARY KEY,
   gym CHAR(20),
   FOREIGN KEY (gym) REFERENCES gym (name)
);

CREATE TABLE go_to
(
   customer CHAR(11),
   gym CHAR(20),
   PRIMARY KEY (customer, gym),
   FOREIGN KEY (customer) REFERENCES customer (SSN),
   FOREIGN KEY (gym) REFERENCES gym (name)
);

CREATE TABLE guest
(
   name CHAR(20),
   age INT,
   customer CHAR(11),
   PRIMARY KEY (customer, name, age),
   FOREIGN KEY (customer) REFERENCES customer (SSN)
);

CREATE TABLE work_at
(
   gym CHAR(20) NOT NULL,
   employee CHAR(11) NOT NULL,
   percetage REAL NOT NULL,
   work_hour CHAR(20) NOT NULL,
   PRIMARY KEY (gym, employee, percetage),
   FOREIGN KEY (gym) REFERENCES gym (name),
   FOREIGN KEY (employee) REFERENCES employee (SSN)
);

CREATE TABLE certifications
```

```
(
   employee CHAR(11) NOT NULL,
   certification_name CHAR(20) NOT NULL,
   PRIMARY KEY (employee, certification_name),
   FOREIGN KEY (employee) REFERENCES employee (SSN)
);
```

Question 2

(1)

```
SELECT S.sname
FROM Suppliers S
WHERE NOT EXISTS((SELECT P.pid
                  FROM Parts P)
                  EXCEPT
                   (SELECT C.pid
                     FROM Catalog C
                     WHERE C.sid = S.sid));
```

(2)

```
SELECT DISTINCT C.sid
FROM Catalog C
WHERE C.cost > (SELECT AVG(C1.cost)
                FROM Catalog C1
                WHERE C1.pid = C.pid);
```

(3)

```
SELECT P.pid, S.sname
FROM Parts P, Suppliers S, Catalog C
WHERE P.pid = C.pid AND C.sid = S.sid
      AND C.cost >= ALL(SELECT C1.cost
                        FROM Catalog C1
                        WHERE C1.pid = C.pid);
```

(4)

```
SELECT C.sid
FROM Catalog C
WHERE NOT EXISTS(SELECT P.color
                 FROM Parts P
                 WHERE P.color <> “red”
                       AND P.pid = C.pid);
```

(5)

```
SELECT C.sid
FROM Catalog C
WHERE NOT EXISTS(SELECT P.color
                 FROM Parts P
                 WHERE (P.color = “red” OR P.color = “green”)
                       AND P.pid = C.pid);
```

(6)

```
SELECT S.sname, MAX(C.cost)
FROM Suppliers S, Catalog C, Parts P
WHERE S.sid = C.sid, C.pid = P.pid
      AND P.color IN (“red”, “green”);
```

Question 3

(1)

```
SELECT M.MovieName
FROM Movies M, MovieSupplier MS, Suppliers S
WHERE (S.SupplierName = "Ben's Video" OR S.SupplierName = "Video Clubhouse")
      AND M.MovieID = MS.MovieID AND S.SupplierID = MS.SupplierID;
```

(2)

```
SELECT M.MovieName
FROM Movies M, Rentals R, Inventory I
WHERE M.MovieID = I.MovieID AND I.TapeID = R.TapeID
      AND R.Duration >= ALL(SELECT Duration FROM Rentals);
```

(3)

```
SELECT S.SupplierName
FROM Supplier S
WHERE S.SupplierID NOT IN
       (SELECT MS.SupplierID
         FROM MovieSupplier MS, Inventory I
         WHERE NOT EXISTS
               (SELECT *
                 FROM MovieSupplier MS2, Inventory I2
                 WHERE MS2.MOVIEID = I2.MovieID
                       AND I2.MovieID = I.MovieID
                       AND MS2.SupplierID = MS.SupplierID);
```

(4)

```
SELECT S.Suppliername, COUNT(DISTINCT MovieID)
FROM Suppliers S, MovieSupplier MS, Inventory I
WHERE I.MovieID = MS.MovieID
       AND S.SupplierID = MS.SupplierID;
```

(5)

```
SELECT M.MovieName
FROM Movie M, Orders O
WHERE M.MovieID = O.MovieID
Group BY M.MovieName
Having SUM(O.Copies) > 4;
```

(6)

```
SELECT C.LastName, C.FirstName
FROM Customers C, Rentals R, Inventory I, Movies M
WHERE C.CustID = R.CustomerID AND R.TapeID = I.TapeID
```

```
    AND I.MovieID = M.MovieID AND M.MovieName = "Kung Fu Panda"
UNION
SELECT C.LastName, C.FirstName
FROM Customers C, Rentals R, Inventory I, MovieSupplier MS, Suppliers S
WHERE C.CustID = R.CustomerID AND R.TapeID = I.TapeID
    AND I.MovieID = MS.MovieID AND MS.SupplierID = S.SupplierID
    AND S.SupplierName = "Palm Video"
(7)
SELECT M.MovieName
FROM Movies M, Inventory I1, Inventory I2
WHERE I1.TapeID <> I2.TapeID AND I1.MovieID = I2.MovieID
    AND I1.MovieID = M.MovieID
(8)
SELECT C.LastName, C.FirstName
FROM Customers C, Rentals R
WHERE C.CustID = R.CustomerID AND R.Duration >= 5
(9)
SELECT S.SupplierName
FROM Suppliers S, MovieSupplier MS, Movies M
WHERE S.SupplierID = MS.SupplierID AND M.MovieName = "Cinderella 2015"
    AND M.MovieID = MS.MovieID
    AND MS.Price <= ALL(SELECT Price
                        FROM MovieSupplier MS, Movies M
                        WHERE MS.MovieID = M.MovieID
                            AND M.MovieName = "Cinderella 2015")
(10)
SELECT MovieName
FROM Movies
WHERE Movies.MovieID NOT IN
    (SELECT MovieID
    FROM Inventory)
```

Question 4

(a)

When it is 4 to 3, the first trigger sets new price to 1.5. Then the second trigger sets new price to 0.75. After that, the following trigger won't work, so the recursion work starts. The outputs are 4 to 0.75, then 4 to 1.5 and finally 4 to 3.

```
UPDATE Purchase
SET price = 1.5
WHERE purchaseID = 111

UPDATE Purchase
SET price = 0.75
WHERE purchaseID = 111
```

- -TODO

(b)

When it is 4 to 3, the first trigger sets new price to 1.5. Then the second trigger sets new price to 0.75. After that, the following trigger won't work. The outputs are 4 to 3, 3 to 1.5 and finally 1.5 to 0.75.

```
UPDATE Purchase
SET price = 1.5
WHERE purchaseID = 111

UPDATE Purchase
SET price = 0.75
WHERE purchaseID = 111
```

- -TODO

(c)

When it is 4 to 3, the first trigger sets new price to 1.5. Then in 4 to 1.5, the second trigger sets new price 0.75. Then it is 4 to 0.75, the following trigger doesn't work. The output won't change.

```
UPDATE Purchase
SET price = 1.5
WHERE purchaseID = 111

UPDATE Purchase
SET price = 0.75
WHERE purchaseID = 111
```

- -TODO