Database Management Fall 2023 HW1

An-Che, Liang B11705009

Problem 1

(a)

The SQL query is as follows:

```sql
SELECT Ssn , Sex , Salary FROM EMPLOYEE
ORDER BY Salary, Ssn
```

Which return the data:

Ssn	Sex	Salary
453453453	F	25000
987987987	M	25000
999887777	F	25000
123456789	M	30000
666884444	M	38000
333445555	M	40000
987654321	F	43000
888665555	M	55000

(b)

The SQL query is as follows:

```sql
SELECT Fname,
    Lname,
    DATE(Bdate, 'auto')
FROM EMPLOYEE
WHERE Fname LIKE "J%"
```

Which return the data:

Fname	Lname	DATE(Bdate, 'auto')
John	Smith	1965-01-08
Jennifer	Wallace	1941-06-19
Joyce	English	1972-07-30
James	Borg	1937-11-09

(c)

The SQL query is as follows:

```sql
SELECT
    E.SSN,
    E.Sex,
    E.Dno,
    D.Dname
FROM
    EMPLOYEE E
JOIN
    DEPARTMENT D ON E.Dno = D.Dnumber
WHERE
    E.Salary <= 30000;
```

Which return the data:

Ssn	Sex	Dno	Dname
123456789	M	5	Research
999887777	F	4	Administration
453453453	F	5	Research
987987987	M	4	Administration

(d)

The SQL query is as follows:

```sql
SELECT E.Ssn,
    COUNT(DISTINCT WO.Pno) AS project_count,
    SUM(WO.Hours) AS work_hours
FROM EMPLOYEE E
    LEFT JOIN WORKS_ON WO ON WO.Essn = E.Ssn
GROUP BY E.Ssn
```

```
HAVING COUNT(DISTINCT WO.Pno) > 0
```

Which return the data:

Ssn	project_count	work_hours
123456789	2	40
333445555	4	40
453453453	2	40
666884444	1	40
888665555	1	NULL
987654321	2	35
987987987	2	40
999887777	2	40

(e)

The SQL query is as follows:

```
SELECT S.Ssn,
    COUNT(*) as Manage_count
FROM Employee E
    JOIN Employee S ON E.Super_ssn = S.Ssn
GROUP BY E.Super_ssn
```

Which return the data:

Ssn	Manage_count
333445555	3
888665555	2
987654321	2

Problem 2

(a)

We can modify the original SQL query into the following:

```
SELECT P.Pnumber
FROM PROJECT P
    JOIN WORKS_ON WO ON WO.Pno = P.Pnumber
    JOIN EMPLOYEE E ON E.Ssn = WO.Essn
WHERE E.Lname = "Smith"
```

(b)

We can modify the original SQL query into the following:

```
SELECT D.Dname,
    E.Lname,
    E.Fname,
    P.Pname
FROM DEPARTMENT D
    JOIN EMPLOYEE E ON E.Dno = D.Dnumber
    JOIN WORKS_ON WO ON WO.Essn = E.Ssn
    JOIN PROJECT P ON P.Pnumber = WO.Pno
ORDER BY D.Dname,
    E.Lname,
    E.Fname
```

Problem 3

(a)

Note that the original price of the same product may change through time, thus we create a record **RETAIL_PRODUCT** to record that property.

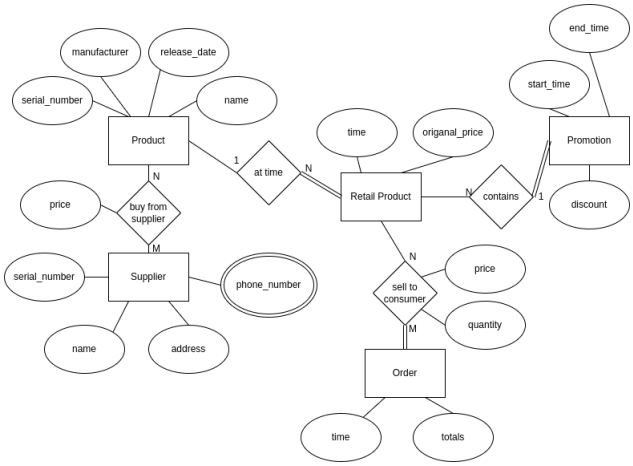

Figure 1: ER Diagram for subproblem (a)

(b)

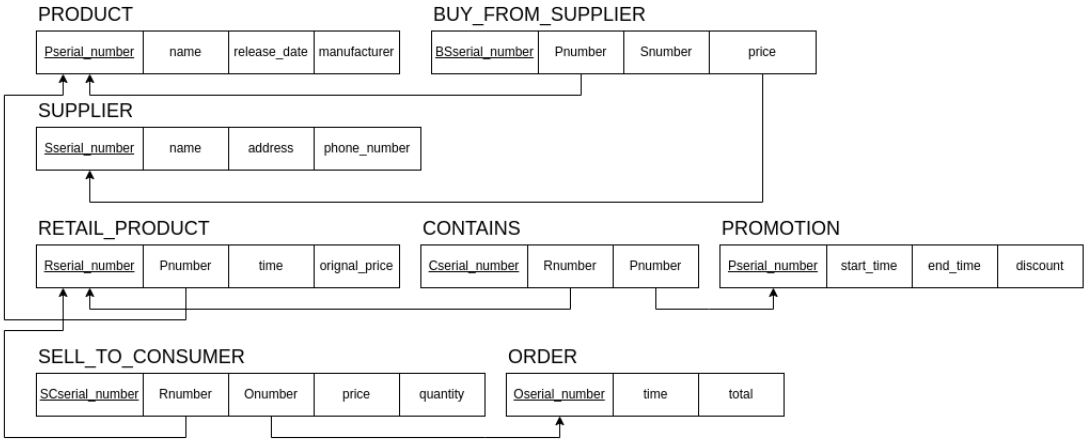

Figure 2: Relational database schema for subproblem (b)

Problem 4

(a)

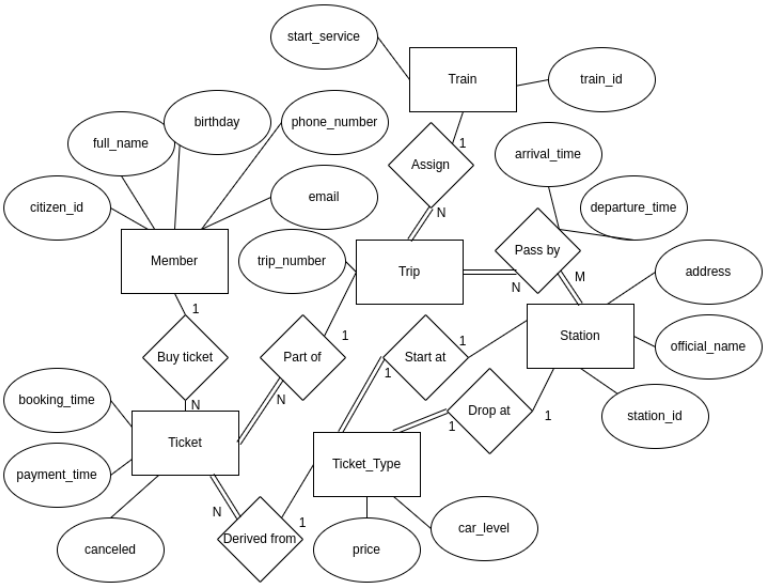

Figure 3: ER Diagram for subproblem (a)

(b)

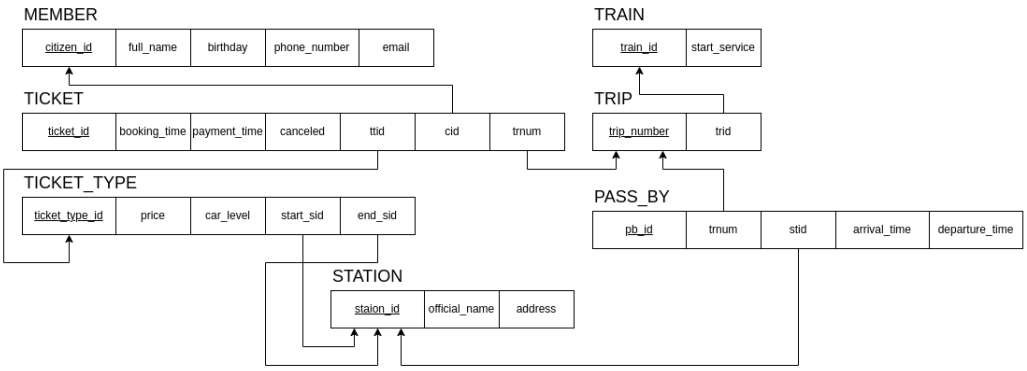

Figure 4: Relational database schema for subproblem (b)

(c)

Then we can construct the database with the following SQL commands, I use `sqlite` as the DBMS.

```sql
DROP TABLE IF EXISTS MEMBER;
CREATE TABLE MEMBER (
    citizen_id VARCHAR(10) NOT NULL UNIQUE PRIMARY KEY,
    full_name VARCHAR(20) NOT NULL UNIQUE,
    birthday DATE NOT NULL,
    phone_number VARCHAR(20) NOT NULL UNIQUE,
    email VARCHAR(100)
);
DROP TABLE IF EXISTS TRAIN;
CREATE TABLE TRAIN(
    train_id BIGINT NOT NULL UNIQUE PRIMARY KEY,
    start_service DATE NOT NULL
);
DROP TABLE IF EXISTS TRIP;
CREATE TABLE TRIP(
    trip_number BIGINT NOT NULL UNIQUE PRIMARY KEY,
    trid BIGINT NOT NULL,
    FOREIGN KEY (trid) REFERENCES TRAIN(train_id)
);
DROP TABLE IF EXISTS STATION;
CREATE TABLE STATION(
    station_id BIGINT NOT NULL UNIQUE PRIMARY KEY,
    official_name VARCHAR(20) NOT NULL UNIQUE,
    address VARCHAR(100) NOT NULL
);
DROP TABLE IF EXISTS PASS_BY;
CREATE TABLE PASS_BY(
    pb_id BIGINT NOT NULL UNIQUE PRIMARY KEY,
    stid BIGINT NOT NULL,
    trnum BIGINT NOT NULL,
    arrival_time TIME NOT NULL,
    departure_time TIME NOT NULL,
    FOREIGN KEY (stid) REFERENCES STATION(station_id),
    FOREIGN KEY (trnum) REFERENCES TRIP(trip_number),
    CHECK(departure_time > arrival_time)
);
DROP TABLE IF EXISTS TICKET_TYPE;
CREATE TABLE TICKET_TYPE (
    ticket_type_id BIGINT NOT NULL UNIQUE PRIMARY KEY,
    price INT NOT NULL,
    car_level INT NOT NULL,
    start_sid BIGINT NOT NULL,
    end_sid BIGINT NOT NULL,
```

```
    FOREIGN KEY (end_sid) REFERENCES STATION(station_id),
    FOREIGN KEY (start_sid) REFERENCES STATION(station_id),
    CHECK(start_sid != end_sid)
);
DROP TABLE IF EXISTS TICKET;
CREATE TABLE TICKET (
    ticket_id BIGINT NOT NULL UNIQUE PRIMARY KEY,
    booking_time TIME NOT NULL,
    payment_time TIME,
    canceled BOOLEAN NOT NULL,
    ttid BIGINT NOT NULL,
    cid VARCHAR(10),
    trnum BIGINT NOT NULL,
    FOREIGN KEY (trnum) REFERENCES TRIP(trip_number),
    FOREIGN KEY (ttid) REFERENCES TICKET_TYPE(ticket_type_id),
    FOREIGN KEY (cid) REFERENCES MEMBER(citizen_id),
    CHECK(payment_time >= booking_time)
);
```