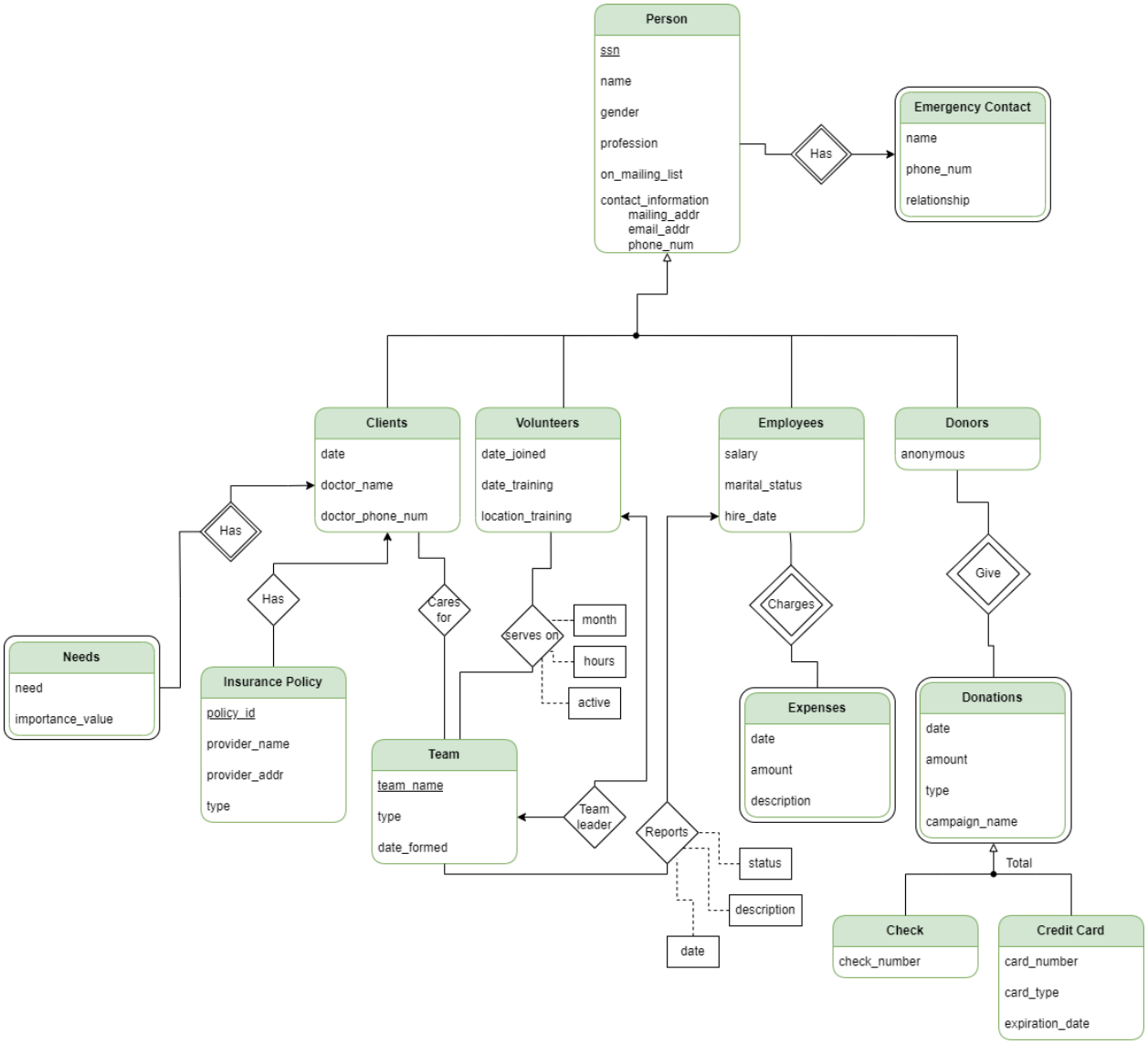
Patient Assistant Network Database System

Kaitlyn Peters
ID: 113558507
kaitlynrpeters@ou.edu

CS/DSA 4513-001
Fall 2024
Dr. Le Gruenwald

Tasks Performed **Page Number**

Task 1. ER Diagram



Task 2. Relational Database Schemas

EmergencyContact(ssn, cname, phone_num, relationship)
Clients(ssn, pname, gender, profession, on_mailing_list, mailing_addr, email_addr, phone_num, date_joined, doctor_name, doctor_phone_num)
Needs(ssn, importance_value)
InsurancePolicy(policy_id, provider_name, provider_addr, type)
Volunteers(ssn, pname, gender, profession, on_mailing_list, mailing_addr, email_addr, phone_num, date_joined, date_training, location_training)
Team(team_name, type, date_formed)
Employees(ssn**,** pname, gender, profession, on_mailing_list, mailing_addr, email_addr, phone_num, salary, marital_status, hire_date)
Expenses(ssn, date, amount, description)
Donors(ssn, pname, gender, profession, on_mailing_list, mailing_addr, email_addr, phone_num, anonymous)
Donations(ssn, date, amount, type, campaign_name)
Check(ssn, check_number)
CreditCard(ssn, card_number, card_type, expiration_date)
ServesOn(ssn, team_name, month, hours, active)
Reports(ssn, team_name, date, description, status)
CaresFor(ssn, team_name)
Leads(team_nanme, ssn)

Task 3.

Task 3.1 Discussion of storage structures for tables

Table Name	Query and Type	Search Key	Query Frequency	Selected File Organization	Justifications
Team	1. Insert 2. Insert 3. Insert 4. Insert 5. Insert 10. Random search 11. Range search 14. Insert	 team_name ssn ssn	1/month 1/week 2/month 2/month 1/year 4/year 1/month 1/year	Heap file	Since we are mostly using insert queries, a heap file will work for this purpose.
Client	1. Insert 2. Insert 8. Random search 10. Random search 15. Delete	 ssn ssn ssn	1/week 1/week 1/week 4/year 1/year	B+ Tree	Supports ordered access with the insert, delete, and search queries, and provides fast searching.
Volunteer	3. Insert 4. Insert 10. Random search	 ssn	2/month 30/month 4/year	B+ Tree	This structure supports efficient access to data despite inserts and searches, thus is well suited for this table.
Employee	5. Insert 6. Insert 9. Range search 13. Range search 14. Insert	 ssn ssn ssn	1/year 1/year 1/month 1/week 1/year	B+ Tree	Similar to client, employee requires many different queries so B+ tree can offer flexibility to inserting and searching.
Expense	6. Insert 9. Range search	 ssn	1/year 1/month	B+ Tree	B+ tree due to the range search query so that it is

					faster to find data with constraints.
Donor	7. Insert 13. Range search	ssn	1/day 1/week	B+ Tree	Similar to employees, this table requires a lot of searching and inserting queries so B+ will offer the structure to easily employ these queries.
Donations	7. Insert 13. Range search	ssn	1/day 1/week	B+ Tree	I choose B+ tree simply for the range search, as it will allow easier access to searching on a constraint.
Person	12. Random search	ssn	1/week	Multitable clustering file organization	Query 12 requires pulling information from multiple tables (donors, employees, etc.) so this structure is best suited for this kind of search.

Task 3.2 Discussion of storage structures for tables (Azure SQL Database)

Initially, I had a table created for a Person, which was a generalization for Client, Volunteer, Donors, and Employees. For Azure SQL, I had to remove the Person table due to how complicated it made the SQL statements. So for the tables, each Client, Volunteer, Donor and Employee have the attributes of person.

For emergency contact, I changed the structure of its table by giving it four foreign keys that linked to social security numbers for each person type (Client, volunteer, donors, and employees), due to how I changed the person table in the database.

Task 4. SQL statements and screenshots showing the creation of tables in Azure SQL Database

4.1 SQL Statements

```sql
-- Kaitlyn Peters
-- DBMS Project 2024
-- TASK 4 -- creete the tables

CREATE TABLE Volunteers(
    ssn VARCHAR(9) PRIMARY KEY,
    pname VARCHAR(100),
    gender VARCHAR(10),
    profession VARCHAR(50),
    on_mailing_list VARCHAR(1), --either yes or no
    mailing_addr VARCHAR(100),
    email_addr VARCHAR(100),
    phone_num VARCHAR(50),
    date_joined DATE,    --make these a date object to store easier
    date_training DATE,  --make these a date object to store easier
    location_training VARCHAR(256),
);

CREATE TABLE Team(
    team_name VARCHAR(100) PRIMARY KEY,
    team_type VARCHAR(100),
    date_formed DATE,
);

CREATE TABLE Employees(
    ssn VARCHAR(9) PRIMARY KEY,
    pname VARCHAR(100),
    gender VARCHAR(10),
    profession VARCHAR(50),
    on_mailing_list VARCHAR(1), --either yes or no
    mailing_addr VARCHAR(100),
    email_addr VARCHAR(100),
    phone_num VARCHAR(50), --no dashes
    salary INTEGER,
    marital_status VARCHAR(50),
    hire_date DATE,
);

CREATE TABLE Expenses(
    ssn VARCHAR(9),
    expense_date DATE,
```

```sql
    expense_amount DECIMAL(10,2),
    expense_description VARCHAR(255),
    FOREIGN KEY(ssn) references Employees,
);

CREATE TABLE Clients(
    ssn VARCHAR(9) PRIMARY KEY,
    pname VARCHAR(100),
    gender VARCHAR(10),
    profession VARCHAR(50),
    on_mailing_list VARCHAR(1), --either yes or no
    mailing_addr VARCHAR(100),
    email_addr VARCHAR(100),
    phone_num VARCHAR(50),
    assignment_date DATE,
    doctor_name VARCHAR(256),
    doctor_phone_number VARCHAR(256),
);

CREATE TABLE Needs(
    ssn VARCHAR(9),
    need VARCHAR(255),
    importance_value INTEGER,
    FOREIGN KEY(ssn) REFERENCES Clients(ssn)
);

CREATE TABLE InsurancePolicy(
    policy_ID INTEGER PRIMARY KEY,
    provider_name VARCHAR(100),
    provider_addr VARCHAR(100),
    insurance_type VARCHAR(100)
);


CREATE TABLE Donors(
    ssn VARCHAR(9) PRIMARY KEY,
    pname VARCHAR(100),
    gender VARCHAR(10),
    profession VARCHAR(50),
    on_mailing_list VARCHAR(1), --either yes or no
    mailing_addr VARCHAR(100),
    email_addr VARCHAR(100),
    phone_num VARCHAR(50),
    is_anonymous VARCHAR(1), --represents a boolean
);

CREATE TABLE Donations(
```

```sql
    ssn VARCHAR(9),
    donation_date DATE,
    donation_amount INTEGER,
    donation_type VARCHAR(50),
    campaign_name VARCHAR(255),
    FOREIGN KEY(ssn) references Donors(ssn),
);

CREATE TABLE Checks(
    ssn VARCHAR(9),
    card_number INTEGER,
    card_type VARCHAR(50),
    FOREIGN KEY(ssn) references Donors(ssn)
);

CREATE TABLE CreditCard(
    ssn VARCHAR(9),
    card_number INTEGER,
    card_type VARCHAR(50),
    expiration_date DATE,
    FOREIGN KEY(ssn) references Donors(ssn)
);

-- ENTITY TABLES
CREATE TABLE EmergencyContact(
    donor_ssn VARCHAR(9),
    volunteer_ssn VARCHAR(9),
    client_ssn VARCHAR(9),
    employee_ssn VARCHAR(9),
    cname VARCHAR(256) PRIMARY KEY,
    contact_phone_number VARCHAR(256),
    relationship VARCHAR(256)
    FOREIGN KEY(donor_ssn) references Donors(ssn),
    FOREIGN KEY(volunteer_ssn) references Volunteers(ssn),
    FOREIGN KEY(client_ssn) references Clients(ssn),
    FOREIGN KEY(employee_ssn) references Employees(ssn)
);


-- relationship tables
-- volunteers <--> team
CREATE TABLE ServesOn(
    ssn VARCHAR(9),
    team_name VARCHAR(100),
    serve_month VARCHAR(100), -- stores the month as a string
    serve_hours INTEGER,
    active VARCHAR(255)
```

```sql
    FOREIGN KEY(ssn) REFERENCES Volunteers(ssn),
    FOREIGN KEY(team_name) REFERENCES Team(team_name)
);

-- clients <--> team
CREATE TABLE Reports(
    ssn VARCHAR(9),
    team_name VARCHAR(100),
    report_status VARCHAR(255),
    report_description VARCHAR(255),
    report_date DATE,
    FOREIGN KEY(ssn) REFERENCES Employees(ssn),
    FOREIGN KEY (team_name) REFERENCES Team(team_name)
);

-- clients <--> team
CREATE TABLE CaresFor(
    ssn VARCHAR(9),
    team_name VARCHAR(100),
    FOREIGN KEY (ssn) REFERENCES Clients(ssn),
    FOREIGN KEY (team_name) REFERENCES Team(team_name)
)

-- a volunteer <--> team
CREATE TABLE Leads(
    ssn VARCHAR(9),
    team_name VARCHAR(100),
    FOREIGN KEY(ssn) REFERENCES Volunteers(ssn),
    FOREIGN KEY(team_name) REFERENCES Team(team_name)
);

-- clients <--> insurance policy
CREATE TABLE Has(
    ssn VARCHAR(9),
    policy_ID INTEGER,
    FOREIGN KEY(ssn) REFERENCES Clients(ssn),
    FOREIGN KEY(policy_ID) REFERENCES InsurancePolicy(policy_ID)
);
```

4.2 Execution Results

project1.sql

Run Cancel Disconnect Change Database: cs-dsa-4513-sql-db Estimated Plan

Enable Actual Plan Parse

```sql
-- Kaitlyn Peters
-- DBMS Project 2024
-- TASK 4 -- creete the tables

-- ENTITY TABLES
CREATE TABLE EmergencyContact(
    cname VARCHAR(256) PRIMARY KEY,
    contact_phone_number VARCHAR(256),
    relationship VARCHAR(256)
);

CREATE TABLE Volunteers(
    ssn VARCHAR(9) PRIMARY KEY,
    pname VARCHAR(100),
    gender VARCHAR(10),
    profession VARCHAR(50),
    on_mailing_list VARCHAR(1), --either yes or no
    mailing_addr VARCHAR(100),
    email_addr VARCHAR(100),
    phone_num VARCHAR(50),
    date_joined DATE,    --make these a date object to store easier
    date_training DATE,  --make these a date object to store easier
    location_training VARCHAR(256),
    emergency_contact_name VARCHAR(256),
    FOREIGN KEY(emergency_contact_name) references EmergencyContact(cname)
```

Messages

```
9:04:18 PM    Started executing query at Line 1
              Commands completed successfully.
              Total execution time: 00:00:00.185
```

Fig. 2 – Successfully executing the CREATE TABLE scripts.

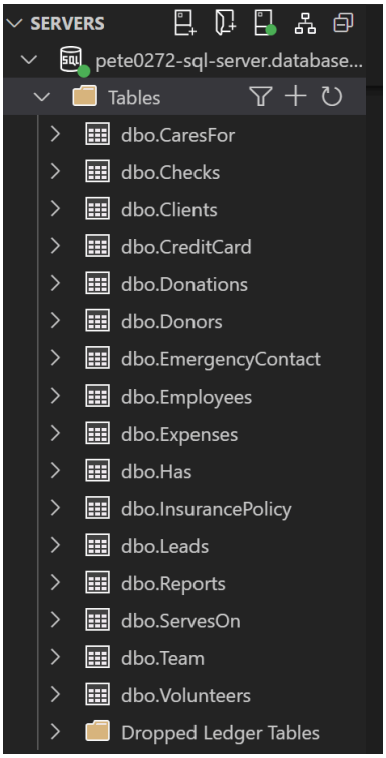


Fig. 3 – Tables shown in the explorer

Task 5. SQL and Java Program

Task 5.1 SQL statements and Transact SQL

Query 1

```
-- 1) Enter a new team into the database (1/month).
DROP PROCEDURE IF EXISTS enterTeam;
GO

CREATE PROCEDURE enterTeam
(
    --team information
    @team_name VARCHAR(100),
    @team_type VARCHAR(100),
    @date_formed DATE
)
AS
BEGIN
    INSERT INTO Team
    VALUES (
        @team_name,
        @team_type,
        @date_formed
    );
END
GO
```

Query 2

```
-- 2) Enter a new client into the database and associate him or her with one or more
teams
-- insert emergency contact to avoid getting errors in the db
DROP PROCEDURE IF EXISTS enterClient;
GO

CREATE PROCEDURE enterClient
(
    --client
    @ssn VARCHAR(9),
    @pname VARCHAR(100),
    @gender VARCHAR(10),
    @profession VARCHAR(50),
    @on_mailing_list VARCHAR(1), --either yes or no
    @mailing_addr VARCHAR(100),
```

```
    @email_addr VARCHAR(100),
    @phone_num VARCHAR(50),
    @assignment_date DATE,
    @doctor_name VARCHAR(256),
    @doctor_phone_number VARCHAR(256),
    --cares for (for the association for a team)
    @team_name VARCHAR(100)
)
AS
BEGIN
    INSERT INTO Clients
    VALUES (
        @ssn,
        @pname,
        @gender,
        @profession,
        @on_mailing_list,
        @mailing_addr,
        @email_addr,
        @phone_num,
        @assignment_date,
        @doctor_name,
        @doctor_phone_number
    );
    INSERT INTO CaresFor
    VALUES(
        @ssn,
        @team_name
    );
END
GO
```

Query 3

```
-- 3) Enter a new volunteer into the database and associate him or her with one or
more teams (2/month).
DROP PROCEDURE IF EXISTS enterVolunteer;
GO

CREATE PROCEDURE enterVolunteer
(
    --volunteer
    @ssn VARCHAR(9),
    @pname VARCHAR(100),
    @gender VARCHAR(10),
```

```
    @profession VARCHAR(50),
    @on_mailing_list VARCHAR(1), --either yes or no
    @mailing_addr VARCHAR(100),
    @email_addr VARCHAR(100),
    @phone_num VARCHAR(50),
    @date_joined DATE,
    @date_training DATE,
    @location_training VARCHAR(256),
    --associate with team (ServesOn)
    @team_name VARCHAR(100),
    @serve_month VARCHAR(100),
    @serve_hours INTEGER,
    @active VARCHAR(255)
)
AS
BEGIN
    INSERT INTO Volunteers
    VALUES(
        @ssn,
        @pname,
        @gender,
        @profession,
        @on_mailing_list,
        @mailing_addr,
        @email_addr,
        @phone_num,
        @date_joined,
        @date_training,
        @location_training
    )
    INSERT INTO ServesOn
    VALUES(
        @ssn,
        @team_name,
        @serve_month,
        @serve_hours,
        @active
    )
END
GO
```

Query 4

```
-- 4) Enter the number of hours a volunteer worked this month for a particular team
(30/month).
DROP PROCEDURE IF EXISTS insertNumberHoursWorked;
```

```sql
GO

CREATE PROCEDURE insertNumberHoursWorked
(
    --servesOn
    @ssn VARCHAR(9),
    @team_name VARCHAR(100),
    @serve_month VARCHAR(255),
    @serve_hours INTEGER,
    @active VARCHAR(255)
)
AS
BEGIN
    INSERT INTO ServesOn
    Values(
        @ssn,
        @team_name,
        @serve_month,
        @serve_hours,
        @active
    )
END
GO
```

Query 5

```sql
-- 5) Enter a new employee into the database and associate him or her with one or more
teams (1/year).
DROP PROCEDURE IF EXISTS enterEmployees;
GO

CREATE PROCEDURE enterEmployees
(
    --employees
    @ssn VARCHAR(9),
    @pname VARCHAR(100),
    @gender VARCHAR(10),
    @profession VARCHAR(50),
    @on_mailing_list VARCHAR(1), --either yes or no
    @mailing_addr VARCHAR(100),
    @email_addr VARCHAR(100),
    @phone_num VARCHAR(50), --no dashes
    @salary INTEGER,
    @marital_status VARCHAR(50),
    @hire_date DATE,
    --reports to
```

```
    @team_name VARCHAR(100),
    @report_status VARCHAR(255),
    @report_description VARCHAR(255),
    @report_date DATE

)
AS
BEGIN
    INSERT INTO Employees
    VALUES(
        @ssn,
        @pname,
        @gender,
        @profession,
        @on_mailing_list,
        @mailing_addr,
        @email_addr,
        @phone_num,
        @salary,
        @marital_status,
        @hire_date
    );
    INSERT INTO Reports
    VALUES(
        @ssn,
        @team_name,
        @report_status,
        @report_description,
        @report_date
    );
END
GO
```

Query 6

```
-- 6) Enter an expense charged by an employee (1/day).
DROP PROCEDURE IF EXISTS enterEmployeeExpense;
GO

CREATE PROCEDURE enterEmployeeExpense
(
    --expense
    @ssn INTEGER,
    @expense_date DATE,
    @expense_amount DECIMAL(10,2),
    @expense_description VARCHAR(255)
)
```

```
AS
BEGIN
    INSERT INTO Expenses
    VALUES(
        @ssn,
        @expense_date,
        @expense_amount,
        @expense_description
    );
END
GO
```

Query 7

```
-- 7) Enter a new donor and associate him or her with several donations (1/day).
DROP PROCEDURE IF EXISTS enterDonorAndDonations;
GO

CREATE PROCEDURE enterDonorAndDonations
(
    --donor
    @ssn VARCHAR(9),
    @pname VARCHAR(100),
    @gender VARCHAR(10),
    @profession VARCHAR(50),
    @on_mailing_list VARCHAR(1),
    @mailing_addr VARCHAR(100),
    @email_addr VARCHAR(100),
    @phone_num VARCHAR(50),
    @is_anonymous VARCHAR(1),
    --donation
    @donation_date DATE,
    @donation_amount INTEGER,
    @donation_type VARCHAR(50),
    @campaign_name VARCHAR(255)
)
AS
BEGIN
    --update donor
    INSERT INTO Donors
    VALUES (
        @ssn,
        @pname,
        @gender,
        @profession,
        @on_mailing_list,
        @mailing_addr,
```

```
        @email_addr,
        @phone_num,
        @is_anonymous
    );
    --associate with a donation
    INSERT INTO Donations
    VALUES(
        @ssn,
        @donation_date,
        @donation_amount,
        @donation_type,
        @campaign_name
    );
END
GO
```

Query 8

```
-- 8) Retrieve the name and phone number of the doctor of a particular client
(1/week).
DROP PROCEDURE IF EXISTS retrieveDoctorInformation;
GO

CREATE PROCEDURE retrieveDoctorInformation
(
    @ssn INTEGER
)
AS
BEGIN
    SELECT doctor_name, doctor_phone_number
    FROM Clients where ssn = @ssn;
END
GO
```

Query 9

```
-- 9) Retrieve the total amount of expenses charged by each employee for a particular
period of time.
-- The list should be sorted by the total amount of expenses (1/month).
DROP PROCEDURE IF EXISTS totalExpenses;
GO

CREATE PROCEDURE totalExpenses
(
    -- constraints for querying period of time
```

```
    @time_start DATE,
    @time_end DATE
)
AS
BEGIN
    --sums the expenses
    SELECT SUM(expense_amount) as totalExpenses
    FROM Expenses
    WHERE expense_Date BETWEEN @time_start and @time_end
    GROUP BY ssn
    ORDER BY totalExpenses
END
GO
```

Query 10

```
-- 10) Retrieve the list of volunteers that are members of teams that support a
particular client (4/year).
DROP PROCEDURE IF EXISTS retrieveVolunteersOfClient;
GO

CREATE PROCEDURE retrieveVolunteersOfClient
(
    @ssn INT
)
AS
BEGIN
    SELECT vol.pname --get all the names of the volunteers
    FROM Volunteers vol
    JOIN ServesOn serves ON vol.ssn = serves.ssn
    JOIN CaresFor cares ON serves.team_name = cares.team_name
    WHERE cares.ssn = @ssn;
END
GO
```

Query 11

```
-- 11) Retrieve the names of all teams that were founded after a particular date
(1/month).
DROP PROCEDURE IF EXISTS retrieveAllTeams;
GO

CREATE PROCEDURE retrieveAllTeams
(
```

```
    @date_start DATE
)
AS
BEGIN
    SELECT t.team_name
    FROM Team t
    WHERE date_formed > @date_start --get teams formed AFTER start
END
GO
```

Query 12

```
-- 12) Retrieve the names, social security numbers, contact information, and emergency
contact
-- information of all people in the database (1/week).
DROP PROCEDURE IF EXISTS retrieveAllPeople;
GO

CREATE PROCEDURE retrieveAllPeople
AS
BEGIN
    --join the EC and person table for each type of person
    -- get the cname from the emergency contact table
    SELECT e.pname, e.ssn, e.mailing_addr, e.phone_num, e.email_addr, ec.cname
    FROM Employees as e
    JOIN EmergencyContact ec ON e.ssn = ec.employee_ssn
    UNION
    SELECT pname, ssn, mailing_addr, phone_num, email_addr, cname
    From Clients as client
    JOIN EmergencyContact ec ON client.ssn = ec.client_ssn
    UNION
    SELECT pname, ssn, mailing_addr, phone_num, email_addr, ec.cname
    FROM Volunteers as vol
    JOIN EmergencyContact ec ON vol.ssn = ec.volunteer_ssn
    UNION
    SELECT pname, ssn, mailing_addr, phone_num, email_addr, ec.cname
    FROM Donors as donor
    JOIN EmergencyContact ec ON donor.ssn = ec.donor_ssn
END
GO
```

Query 13

```
-- 13) Retrieve the name and total amount donated by donors that are also employees.
The list
-- should be sorted by the total amount of the donations, and indicate if each donor
wishes to
-- remain anonymous (1/week)
DROP PROCEDURE IF EXISTS allDonorsAndEmployees;
GO

CREATE PROCEDURE allDonorsAndEmployees
AS
BEGIN
    SELECT donor.pname, SUM(donation.donation_amount) AS totalDonation,
donor.is_anonymous
    FROM Donors donor
    --get donation amount corresponding to donor
    JOIN Donations donation on donor.ssn = donation.ssn
    WHERE donor.ssn IN (SELECT ssn FROM Employees)
    GROUP BY donor.pname, donor.is_anonymous
    -- sort by total donation
    ORDER BY totalDonation
END;
GO
```

Query 14

```
-- 14) Increase the salary by 10% of all employees to whom more than one team must
report. (1/year)
DROP PROCEDURE IF EXISTS increaseSalary;
GO

CREATE PROCEDURE increaseSalary
AS
BEGIN
    UPDATE Employees
    SET salary = salary * 1.1
    WHERE ssn in (SELECT ssn from reports GROUP BY ssn HAVING COUNT(team_name) > 1);
END;
GO
```

Query 15

```
-- 15) Delete all clients who do not have health insurance and whose value of
importance for
```

```sql
-- transportation is less than 5 (4/year).
DROP PROCEDURE IF EXISTS deleteClientsWithoutInsurnace;
GO

CREATE PROCEDURE deleteClientsWithoutInsurnace
AS
BEGIN
    DELETE FROM Clients
    WHERE ssn NOT IN (SELECT ssn FROM InsurancePolicy WHERE insurance_type = 'Health')
    AND ssn IN (SELECT ssn FROM Needs WHERE need = 'Transportantion'
        AND importance_value < 5);
END;
GO
```

Query 16

```sql
-- 16) Import: enter new teams from a data file until the file is empty (the user must
be asked
-- to enter the input file name).
-- opens the file in the java and then inserts each line using
-- this sql procedure
DROP PROCEDURE IF EXISTS importTeams
GO

CREATE PROCEDURE importTeams
(
    @team_name VARCHAR(100),
    @team_type VARCHAR(100),
    @date_formed DATE
)
AS
BEGIN
    INSERT INTO Team
    VALUES(
        @team_name,
        @team_type,
        @date_formed
    )
END;
GO
```

Query 17

```
-- 17) export retrieve names and mailing addresses of all peple on the mamiling list
and output them to a data file
-- going to use a csv file to export
DROP PROCEDURE IF EXISTS exportMailingList;
GO

CREATE PROCEDURE exportMailingList
AS
BEGIN
    SELECT pname, mailing_addr
      FROM Clients
     WHERE UPPER(on_mailing_list) = 'Y'
     UNION
    SELECT pname, mailing_addr
      FROM Volunteers
     WHERE UPPER(on_mailing_list) = 'Y'
     UNION
    SELECT pname, mailing_addr
      FROM Employees
     WHERE UPPER(on_mailing_list) = 'Y'
     UNION
    SELECT pname, mailing_addr
      FROM Donors
     WHERE UPPER(on_mailing_list) = 'Y'
END;
GO
```

Query 18

* Done in Java

```
case "18":
                    System.out.println("Closing the PAN Database system...");
                    sc.close(); // Close the scanner before exiting the
application
                    conn.close(); //Close the DB connection
                    System.exit(0); //exit the program
                    break;
```

Task 5.2 The Java source program and screenshots showing its successful compilation

5.2.1 Java Source Program

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.sql.CallableStatement;
import java.sql.Connection;
import java.sql.Date;
import java.sql.Statement;
import java.util.Scanner;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
/*
* Make connections to the database
*/
public class PAN{
        // Database credentials
   final static String HOSTNAME = "pete0272-sql-server.database.windows.net";
   final static String DBNAME = "cs-dsa-4513-sql-db";
   final static String USERNAME = "pete0272";
   final static String PASSWORD = "Ambergrisb03!";
   // Database connection string
   final static String URL =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt
=true;trustServerCertificate=false;hostNameInCertificate=*.database.windows.net
;loginTimeout=30;",
           HOSTNAME, DBNAME, USERNAME, PASSWORD);


   /*
    * string of all the queries to be asked
    */
   final static String PROMPT =
           "\nPlease select one of the options below: \n" +
           "1) Enter a new team into the database \n" +
           "2) Enter a new client into the database and associate him or her
with one or more teams\n" +
           "3) Enter a new volunteer into the database and associate him or her
with one or more teams \n" +
```

```
            "4) Enter the number of hours a volunteer worked this month for a
particular team \n" +
            "5) Enter a new employee into the database and associate him or her
with one or more teams \n" +
            "6) Enter an expense charged by an employee \n" +
            "7) Enter a new donor and associate him or her with several
donations \n" +
            "8) Retrieve the name and phone number of the doctor of a particular
client \n" +
            "9) Retrieve the total amount of expenses charged by each employee
for a particular period of "
            + "time. The list should be sorted by the total amount of expenses
\n" +
            "10) Retrieve the list of volunteers that are members of teams that
support a particular client \n" +
            "11) Retrieve the names of all teams that were founded after a
particular date \n" +
            "12) Retrieve the names, social security numbers, contact
information, and emergency contact "  +
            "information of all people in the database  \n" +
            "13) Retrieve the name and total amount donated by donors that are
also employees. The list "
            + "should be sorted by the total amount of the donations, and
indicate if each donor wishes to "
            + "remain anonymous  \n" +
            "14) Increase the salary by 10% of all employees to whom more than
one team must report \n" +
            "15) Delete all clients who do not have health insurance and whose
value of importance for transportation is less than 5\n" +
            "16) Import: enter new teams from a data file until the file is
empty (the user must be asked" +
            " to enter the input file name). \n" +
            "17) Export: Retrieve names and mailing addresses of all people on
the mailing list and " +
             "output them to a data file instead of screen (the user must be
asked to enter the output file " +
             "name).  \n" +
            "18) Quit \n";

    /*
     * Main function with exception handling
     */
    public static void main(String[] args) throws SQLException, IOException,
FileNotFoundException {
       System.out.println("Welcome to the Patient Assistant Network (PAN)
Database System");

        final Scanner sc = new Scanner(System.in); // Scanner is used to collect
the user input
```

```
String option = ""; // Initialize user option selection as nothing

//create a connection
Connection conn = DriverManager.getConnection(URL);
//loop until user selects query 18 to QUIT
while (!option.equals("18")) {
     System.out.println(PROMPT); // print the available queries
     option = sc.next(); //get user input
     //prompt for queries
    switch (option) {
           case "1":
           enterTeam(conn, sc);
           break;

           case "2":
           enterClient(conn, sc);
           break;

           case "3":
           enterVolunteer(conn, sc);
           break;

           case "4":
           insertNumberHoursWorked(conn, sc);
           break;

           case "5":
           enterEmployees(conn, sc);
           break;

           case "6":
           enterEmployeeExpense(conn, sc);
           break;

           case "7":
           enterDonorAndDonations(conn, sc);
           break;

           case "8":
           retrieveDoctorInformation(conn, sc);
           break;

           case "9":
           totalExpenses(conn, sc);
           break;

           case "10":
           retrieveVolunteersOfClient(conn, sc);
           break;
```

```
                    case "11":
                    retrieveAllTeams(conn, sc);
                    break;

                    case "12":
                    retrieveAllPeople(conn, sc);
                    break;

                    case "13":
                    allDonorsAndEmployees(conn, sc);
                    break;

                    case "14":
                    increaseSalary(conn, sc);
                    break;

                    case "15":
                    deleteClientsWithoutInsurance(conn, sc);
                    break;

                    case "16":
                    importData(conn, sc);
                    break;

                    case "17":
                    exportData(conn, sc);
                    break;

                    case "18":
                    System.out.println("Closing the PAN Database system...");
                    sc.close(); // Close the scanner before exiting the
application
                    conn.close(); //Close the DB connection
                    System.exit(0); //exit the program
                    break;


            }
        }
    }


    /*
     * query 1
     */
    public static void enterTeam(Connection conn, Scanner sc) throws
SQLException{
        String input1, input2, input3;
```

```
      System.out.println("Enter the team name: \n");
      input1 = sc.next();

      System.out.println("Enter the team type: \n");
      input2 = sc.next();

      System.out.println("Enter the date that the team formed (YYYY-MM-DD):
\n");
      input3 = sc.next();
      Date date = java.sql.Date.valueOf(input3); //convert to a date object



      //create the connection
      CallableStatement stmt = conn.prepareCall("{call enterTeam(?, ?, ?)}");
      stmt.setString(1, input1);
      stmt.setString(2, input2);
      stmt.setDate(3, date);

      stmt.execute();
   }

   /*
    * query 2
    */
   public static void enterClient(Connection conn, Scanner sc) throws
SQLException{
//     String cname, cphone, relationship;
//     String ssn, pname, gender, profession, mailingList, mailingAddr,
emailAddr, phone,
//            assignmentDate, doctorName, doctorPhone;
//     String team_name;
      String input;
      int intInput;

      //configure the connection string
      CallableStatement stmt = conn.prepareCall("{call enterClient(?, ?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?)}");

      //-----------------
      //ssn
      System.out.println("Enter the client's social security number:\n");
      input = sc.next();
      stmt.setString(1, input);

      //pname
      System.out.println("Enter the client's name");
      input = sc.next();
```

```
stmt.setString(2, input);

//gender
System.out.println("Enter the client's gender:\n");
input = sc.next();
stmt.setString(3, input);

//profession
System.out.println("Enter the client's profession:\n");
input = sc.next();
stmt.setString(4, input);

//on_mailing_list
System.out.println("Is the client on the mailing list? (Y or N)\n");
input = sc.next();
stmt.setString(5, input);

//mailing_addr
System.out.println("Enter the client's mailing address:\n");
input = sc.next();
stmt.setString(6, input);

//email_addr
System.out.println("Enter the client's email address:\n");
input = sc.next();
stmt.setString(7, input);

//phone number
System.out.println("Enter the client's phone number:\n");
input = sc.next();
stmt.setString(8, input);

//assignment date
System.out.println("Enter the client's assignment date (YYYY-MM-DD):\n");
input = sc.next();
Date date = java.sql.Date.valueOf(input); //convert to a date object
stmt.setDate(9, date);

//doctor name
System.out.println("Enter the client's doctor's name:\n");
input = sc.next();
stmt.setString(10, input);

//doctor phone number
System.out.println("Enter the client's doctor's phone number: \n");
input = sc.next();
stmt.setString(11, input);

//team name
```

```
        System.out.println("Enter the team name associated with this client:
\n");
        input = sc.next();
        stmt.setString(12, input);

        stmt.execute();
    }

    /*
     * query 3
     */
    public static void enterVolunteer(Connection conn, Scanner sc) throws
SQLException{
        String input;
        int intInput;

        //configure the connection string
        CallableStatement stmt = conn.prepareCall("{call enterVolunteer(?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, ?, ?,"
                    + "?, ?, ?)}");
        /*
         * volunteer information
         */
        //ssn
        System.out.println("Enter the volunteer's social security number:\n");
        input = sc.next();
        stmt.setString(1, input);

        //name
        System.out.println("Enter the volunteer's name:\n");
        input = sc.next();
        stmt.setString(2, input);

        //gender
        System.out.println("Enter the volunteer's gender:\n");
        input = sc.next();
        stmt.setString(3, input);

        //profession
        System.out.println("Enter the volunteer's profession:\n");
        input = sc.next();
        stmt.setString(4, input);

        //mailing list
        System.out.println("Is the volunteer on the mailing list? (Y or N)\n");
        input = sc.next();
        stmt.setString(5, input);

        //mailing addr
```

```
        System.out.println("Enter the volunteer's mailing address:\n");
        input = sc.next();
        stmt.setString(6, input);

        //email
        System.out.println("Enter the volunteer's email address:\n");
        input = sc.next();
        stmt.setString(7, input);

        //phone
        System.out.println("Enter the volunteer's phone number:\n");
        input = sc.next();
        stmt.setString(8, input);

        //date joined
        System.out.println("Enter the volunteer's date joined (YYYY-MM-DD):\n");
        input = sc.next();
        Date date = java.sql.Date.valueOf(input); //convert to a date object
        stmt.setDate(9, date);

        //date training
        System.out.println("Enter the volunteer's date training
(YYYY-MM-DD):\n");
        input = sc.next();
        date = java.sql.Date.valueOf(input); //convert to a date object
        stmt.setDate(10, date);
        //training location
        System.out.println("\nEnter the training location:");
        input = sc.next();
        stmt.setString(11, input);

        /*
         * servesOn information
         */
        //team name
        System.out.println("Enter the team name associated with this volunteer:
\n");
        input = sc.next();
        stmt.setString(12, input);

        //month served
        System.out.println("Enter the month the volunteer served: \n");
        input = sc.next();
        stmt.setString(13, input);

        //serves hours
        System.out.println("Enter the number of hours the volunteer served: \n");
        intInput = sc.nextInt();
        stmt.setInt(14, intInput);
```

```
      //active
      System.out.println("Indicate the status of the volunteer on the team:
\n");
      input = sc.next();
      stmt.setString(15, input);


      stmt.execute(); //carry out query
   }

   /*
    * query 4 -- number of hours worked
    */
   public static void insertNumberHoursWorked(Connection conn, Scanner sc)
throws SQLException{
      String input;
      int intInput;

      //configure the connection string
      CallableStatement stmt = conn.prepareCall("{call
insertNumberHoursWorked(?, ?, ?, ?, ?)}");

      /*
       * enter serves on information
       */
      //ssn
      System.out.println("Enter the volunteer's social security number:\n");
      input = sc.next();
      stmt.setString(1, input);

      //name
      System.out.println("Enter the volunteer's team name:\n");
      input = sc.next();
      stmt.setString(2, input);

      //serve month
      System.out.println("Enter the month the volunteer served: \n");
      input = sc.next();
      stmt.setString(3, input);

      //serve hours
      System.out.println("Enter the month the volunteer served: \n");
      intInput = sc.nextInt();
      stmt.setInt(4, intInput);

      //active
      System.out.println("Is the volunteer active? \n");
      input = sc.next();
```

```
        stmt.setString(5, input);

        stmt.execute(); //carry out query
    }

    /*
     * query 5 --  Enter a new employee into the database and associate him or
her with one or more teams
     */
    public static void enterEmployees(Connection conn, Scanner sc) throws
SQLException{
        //input variables
        String input, emergencycontact;
        Date date;
        int intInput;

        //create connection
        CallableStatement stmt = conn.prepareCall("{call enterEmployees(?, ?,"
                    + "?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)}");

        //-----------------
        //ssn
        System.out.println("Enter the employee's social security number:\n");
        input = sc.next();
        stmt.setString(1, input);

        //pname
        System.out.println("\nEnter the employee's name:");
        input = sc.next();
        stmt.setString(2, input);

        //gender
        System.out.println("\nEnter the employee's gender:");
        input = sc.next();
        stmt.setString(3, input);

        //profession
        System.out.println("\nEnter the employee's profession:");
        input = sc.next();
        stmt.setString(4, input);

        //on_mailing_list
        System.out.println("\nIs the employee on the mailing list? (Y or N):");
        input = sc.next();
        stmt.setString(5, input);

        //mailing_addr
        System.out.println("\nEnter the employee's mailing address:");
        input = sc.next();
```

```
        stmt.setString(6, input);

        //email_addr
        System.out.println("\nEnter the employee's email address:");
        input = sc.next();
        stmt.setString(7, input);

        //phone number
        System.out.println("\nEnter the employee's phone number:");
        input = sc.next();
        stmt.setString(8, input);

        //salary
        System.out.println("Enter the employee's salary:\n");
        intInput = sc.nextInt();
        stmt.setInt(9, intInput);

        //marital status
        System.out.println("\nEnter the employee's marital status:");
        input = sc.next();
        stmt.setString(10, input);

        //hire date
        System.out.println("\nEnter the employee's hire date(YYYY-MM-DD): ");
        input = sc.next();
        date = java.sql.Date.valueOf(input);
        stmt.setDate(11, date);

        /*
         * reports to
         */

        //team name
        System.out.println("\nEnter the team name associated with this
employee:");
        input = sc.next();
        stmt.setString(12, input);

        //report status
        System.out.println("\nEnter the report status:");
        input = sc.next();
        stmt.setString(13, input);

        //report description
        System.out.println("\nEnter the report description:");
        input = sc.next();
        stmt.setString(14, input);

        //report date
```

```
        System.out.println("\nEnter date of the report (YYYY-MM-DD):");
        input = sc.next();
        date = java.sql.Date.valueOf(input);
        stmt.setDate(15, date);

        stmt.execute();
    }

    /*
     * query 6 -- enter employee expenses
     */
    public static void enterEmployeeExpense(Connection conn, Scanner sc) throws
SQLException{
        //input variables
        String input;
        Date date;

        //create connection
        CallableStatement stmt = conn.prepareCall("{call enterEmployeeExpense(?,
?, ?, ?)}");

        //employee ssn
        System.out.println("\nEnter the employee's ssn:");
        input = sc.next();
        stmt.setString(1, input);

        //expense date
        System.out.println("\nEtner the expense date:");
        input = sc.next();
        date = java.sql.Date.valueOf(input);
        stmt.setDate(2, date);

        //expense amount
        System.out.println("\nEnter the expense amount:");
        input = sc.next();
        stmt.setString(3, input);

        //expense description
        System.out.println("\nEnter the expense description:");
        input = sc.next();
        stmt.setString(4, input);

        stmt.execute();
    }


    /*
     * query 7
     */
```

```
   public static void enterDonorAndDonations(Connection conn, Scanner sc)
throws SQLException{
      String input;
      Date date;
      int intInput;

      CallableStatement stmt = conn.prepareCall("{call
enterDonorAndDonations(?, ?, ?, ?, ?,"
                  + "?, ?, ?, ?, ?, ?, ?, ?)}");
      /*
       * donor
       */
      System.out.println("Enter the donor's social security number:\n");
      input = sc.next();
      stmt.setString(1, input);

      //pname
      System.out.println("\nEnter the donor's name:");
      input = sc.next();
      stmt.setString(2, input);

      //gender
      System.out.println("\nEnter the donor's gender:");
      input = sc.next();
      stmt.setString(3, input);

      //profession
      System.out.println("\nEnter the donor's profession:");
      input = sc.next();
      stmt.setString(4, input);

      //on_mailing_list
      System.out.println("\nIs the donor on the mailing list? (Y or N):");
      input = sc.next();
      stmt.setString(5, input);

      //mailing_addr
      System.out.println("\nEnter the donor's mailing address:");
      input = sc.next();
      stmt.setString(6, input);

      //email_addr
      System.out.println("\nEnter the donor's email address:");
      input = sc.next();
      stmt.setString(7, input);

      //phone number
      System.out.println("\nEnter the donor's phone number:");
      input = sc.next();
```

```
        stmt.setString(8, input);

        System.out.println("\nDoes the donor wish to remain anonymous? (Y or
N):");
        input = sc.next();
        stmt.setString(9, input);

        /*
         * donation
         */
        System.out.println("\nEnter date of the donation (YYYY-MM-DD):");
        input = sc.next();
        date = java.sql.Date.valueOf(input);
        stmt.setDate(10, date);

        System.out.println("\nEnter donation amount:");
        intInput = sc.nextInt();
        stmt.setInt(11, intInput);

        System.out.println("\nEnter the type of donation: ");
        input = sc.next();
        stmt.setString(12, input);

        System.out.println("\nEnter the campaign name: ");
        input = sc.next();
        stmt.setString(13, input);

        stmt.execute();
    }

    /*
     * query 8 -- Retrieve the name and phone number of the doctor of a
particular client
     */
    public static void retrieveDoctorInformation(Connection conn, Scanner sc)
throws SQLException{
        String input;

        System.out.println("\nEnter the ssn of the client:");
        input = sc.next();

        CallableStatement stmt = conn.prepareCall("{call
retrieveDoctorInformation(?)}");
        stmt.setString(1,input);

        ResultSet rs = stmt.executeQuery();

        while(rs.next()) {
              System.out.println("Doctor name:"
```

```
                        + rs.getString("doctor_name"));
            System.out.println("Doctor's phone number:"
                        + rs.getString("doctor_phone_number"));
        }
    }

    /*
     * query 9 -- Retrieve the total amount of expenses charged by each employee
for a particular period of time. The list should be sorted by the total amount
of expenses
     */
    public static void totalExpenses(Connection conn, Scanner sc) throws
SQLException{
        String input;

        CallableStatement stmt = conn.prepareCall("{call totalExpenses(?, ?)}");

        System.out.println("Enter the start time for expenses:");
        input = sc.next();
        Date date = java.sql.Date.valueOf(input);
        stmt.setDate(1, date);

        System.out.println("Enter the end time for expenses:");
        input = sc.next();
        date = java.sql.Date.valueOf(input);
        stmt.setDate(2, date);

        ResultSet rs = stmt.executeQuery();

        //start getting the query
        while(rs.next()) {
            System.out.println("Total expenses: " +
rs.getDouble("totalExpenses"));
        }

    }

    /*
     * query 10 -- Retrieve the list of volunteers that are members of teams
that support a particular client
     */
    public static void retrieveVolunteersOfClient(Connection conn, Scanner sc)
throws SQLException{
        String input;

        System.out.println("\nEnter the ssn of the client:");
        input = sc.next();
```

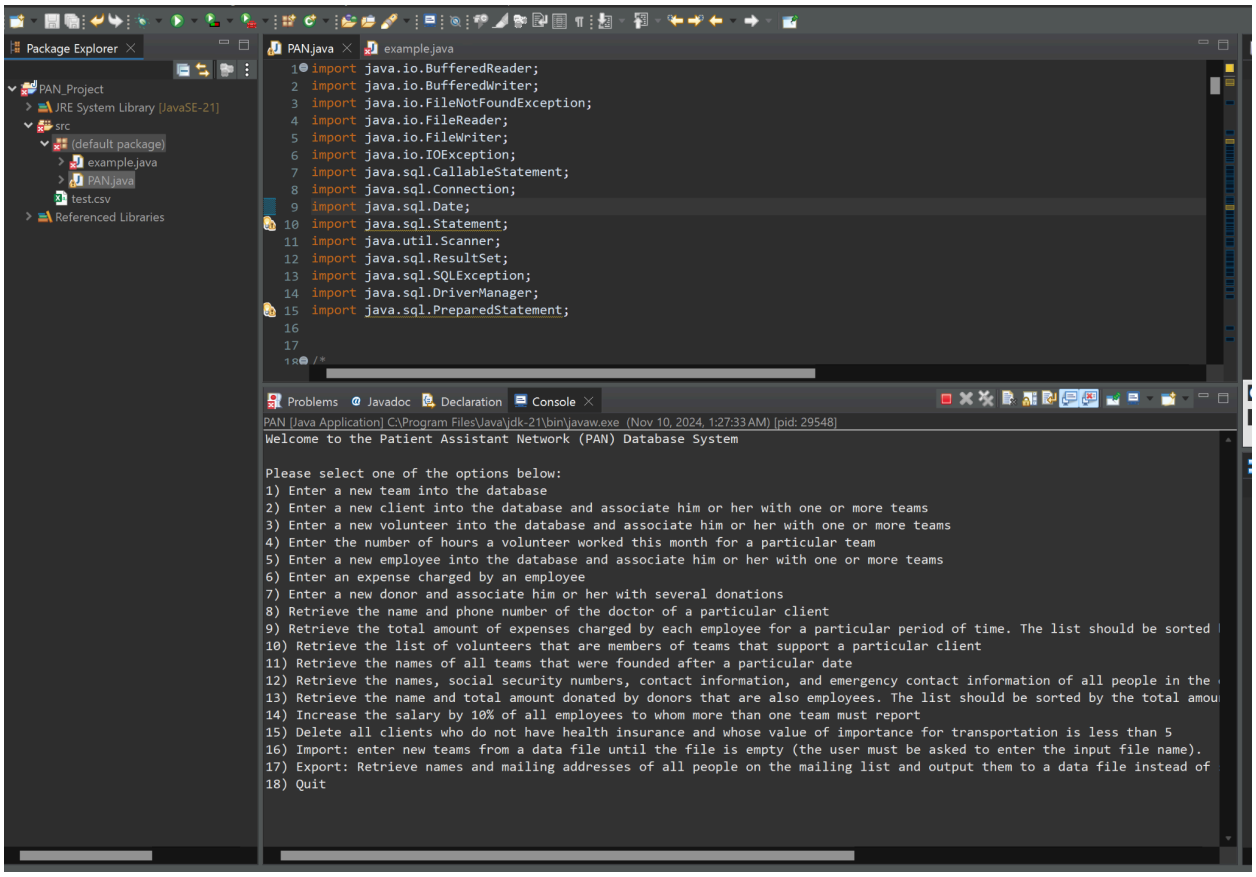
```
      CallableStatement stmt = conn.prepareCall("{call
retrieveVolunteersOfClient(?)}");
      stmt.setString(1,input);

      ResultSet rs = stmt.executeQuery();

      System.out.println("\nVolunteers:");

     //start getting the query
      while(rs.next()) {
            System.out.println(rs.getString("pname"));
      }
   }

   /*
    * query 11
    */
   public static void retrieveAllTeams(Connection conn, Scanner sc) throws
SQLException{

      CallableStatement stmt = conn.prepareCall("{call retrieveAllTeams(?)}");
      System.out.println("Enter a start date");
      String input = sc.next();
      Date date = java.sql.Date.valueOf(input);
      stmt.setDate(1, date);

      //open the connection
      ResultSet rs = stmt.executeQuery();

      System.out.println("\nTeams founded after " + date + ":");

      while(rs.next()) {
            System.out.println(rs.getString("team_name"));
      }
   }

   /*
    * query 12 -- Retrieve the names, social security numbers, contact
information, and emergency contact
       * information of all people in the database (1/week).
    */
   public static void retrieveAllPeople(Connection conn, Scanner sc) throws
SQLException{
      CallableStatement stmt = conn.prepareCall("{call retrieveAllPeople()}");

      ResultSet rs = stmt.executeQuery();

      //prints out a formatted header
      System.out.println("\nAll people in the database:");
```

```
      String formatted = String.format("%-10s %-10s %-20s %-20s %-20s %-20s",
                  "Name", "ssn", "Mailing Address", "Phone Number", "Email
Address",
                  "Emergency Contact Name");

      System.out.println(formatted);

System.out.println("-----------------------------------------------------"
                  + "-----------------------------------------------------"
                  + "-----");

      //print all the information from the query
      while(rs.next()) {
            //formatted string
            formatted = String.format("%-10s %-10s %-20s %-20s %-20s %-20s",
                        rs.getString("pname"), rs.getString("ssn"),
rs.getString("mailing_addr"),
                        rs.getString("phone_num"), rs.getString("email_addr"),
                        rs.getString("cname"));
            //print info
            System.out.println(formatted);
      }
   }

   /*
    * query 13 -- Retrieve the name and total amount donated by donors that are
also employees. The list
       * should be sorted by the total amount of the donations, and indicate if
each donor wishes to
       * remain anonymous (1/week)
    */
   public static void allDonorsAndEmployees(Connection conn, Scanner sc) throws
SQLException{
      CallableStatement stmt = conn.prepareCall("{call
allDonorsAndEmployees()}");
      ResultSet rs = stmt.executeQuery();

     //prints out a formatted header
      System.out.println("\nDonations from doctors that are also employees:");
      String formatted = String.format("%-20s %-20s",
                  "Donor Name", "Total Amount");

      System.out.println(formatted);

System.out.println("-------------------------------------------------------");

      while(rs.next()) {
            formatted = String.format("%-20s %-20s",
                        rs.getString("pname"), rs.getString("totalDonation"));
```

```
            System.out.println(formatted);
        }
    }
    /*
     * query 14 -- Increase the salary by 10% of all employees to whom more than
one team must report. (1/year)
     */
    public static void increaseSalary(Connection conn, Scanner sc) throws
SQLException{
        CallableStatement stmt = conn.prepareCall("{call increaseSalary()}");
        System.out.println("Increasing salary by 1.1% ...");
        stmt.execute();
    }

    /*
     * query 15 -- Delete all clients who do not have health insurance and whose
value of importance for
       * transportation is less than 5 (4/year).
     */
    public static void deleteClientsWithoutInsurance(Connection conn, Scanner
sc) throws SQLException{
        CallableStatement stmt = conn.prepareCall("{call
deleteClientsWithoutInsurance()}");

        System.out.println("Deleting clients without insurance ...");
        stmt.execute();
    }

    /* ----------------------------------------
     * query 16 -- Import: enter new teams from a data file until the file is
empty (the user must be asked
       *    to enter the input file name).
     * ---------------------------------------- */
    public static void importData(Connection conn, Scanner sc) throws
SQLException, IOException, FileNotFoundException{
        String fileName;
        BufferedReader reader;

        System.out.println("\nEnter the file name to load data from:");
        fileName = sc.next();

        //open connection
        CallableStatement stmt = conn.prepareCall("{call importTeams(?, ?, ?)}");

        //open the file
            reader = new BufferedReader(new FileReader("src/" + fileName));
            String line;
            //start reading the data and making a sql query
```

```
            while((line = reader.readLine()) != null) {
                  String[] row = line.split(",");

                  //get each col and execute the query
                  stmt.setString(1, row[0]); //team name
                  stmt.setString(2, row[1]); //team type

                  //get date and remove whitespace
                  String date = (row[2]).replaceAll("\\s+",""); //date founded
                  stmt.setDate(3, java.sql.Date.valueOf(date));
                  stmt.execute();
            }
            reader.close();
   }

   /*
    * query 17 Retrieve names and mailing addresses of all people on the
mailing list and
       * output them to a data file instead of screen (the user must be asked
to enter the output file
       * name).
    */
   public static void exportData(Connection conn, Scanner sc) throws
SQLException, IOException, FileNotFoundException{
      String filename = "test.csv";
      String path = "src/";

      CallableStatement stmt = conn.prepareCall("{call exportMailingList()}");
      ResultSet rs = stmt.executeQuery();

      System.out.println("\nEnter a file name to output to: ");
      filename = sc.next();
      //open file to write to
      BufferedWriter writer = new BufferedWriter(new FileWriter(path +
filename));
      while(rs.next()) {
            //delimit the data with columns for a .csv file
            writer.write(rs.getString("pname") + ","
                        + rs.getString("mailing_addr") + ",");
            writer.write("\n"); //newline
      }
      writer.close();
   }

}
```

5.2.2 Java Compilation



Task 6. Java program Execution

Task 6.1 Screenshots showing the testing of query 1

```
1
Enter the team name:

team1
Enter the team type:

Health
Enter the date that the team formed (YYYY-MM-DD):

2022-01-01
```

```
1
Enter the team name:

team2
Enter the team type:

Heart
Enter the date that the team formed (YYYY-MM-DD):

2023-02-02
```

```
1
Enter the team name:

team3
Enter the team type:

Lung
Enter the date that the team formed (YYYY-MM-DD):

2023-11-02
```

```
1
Enter the team name:

team4
Enter the team type:

Skin
Enter the date that the team formed (YYYY-MM-DD):

2021-01-01
```

```
1
Enter the team name:

team5
Enter the team type:

Eyes
Enter the date that the team formed (YYYY-MM-DD):

2023-03-01
```

Results of queries:

Results | Messages

	team_name	team_type	date_formed
1	team1	Health	2022-01-01
2	team2	Heart	2023-02-02
3	team3	Lung	2023-11-02
4	team4	Skin	2021-01-01
5	team5	Eyes	2023-03-01

Task 6.2 Screenshots showing the testing of query 2

```
2
Enter the client's social security number:

1
Enter the client's name
A
Enter the client's gender:

F
Enter the client's profession:

Teacher
Is the client on the mailing list? (Y or N)

Y
Enter the client's mailing address:

a@gmail.com
Enter the client's email address:

111 Dr
Enter the client's phone number:

444-000-11111
Enter the client's assignment date (YYYY-MM-DD):

2021-02-02
Enter the client's doctor's name:

Dr. J
Enter the client's doctor's phone number:

333-000-0000
Enter the team name associated with this client:

team1
```

```
2
Enter the client's social security number:

2
Enter the client's name
J
Enter the client's gender:

M
Enter the client's profession:

Salesman
Is the client on the mailing list? (Y or N)

N
Enter the client's mailing address:

111 Dr
Enter the client's email address:

j@gmail.com
Enter the client's phone number:

444-000-1111
Enter the client's assignment date (YYYY-MM-DD):

2022-02-02
Enter the client's doctor's name:

Dr. J
Enter the client's doctor's phone number:

333-000-0000
Enter the team name associated with this client:

team1
```

```
2
Enter the client's social security number:

3
Enter the client's name
T
Enter the client's gender:

F
Enter the client's profession:

Doctor
Is the client on the mailing list? (Y or N)

Y
Enter the client's mailing address:

2324 Dr
Enter the client's email address:

t@mail.com
Enter the client's phone number:

333-000-0000
Enter the client's assignment date (YYYY-MM-DD):

2024-01-11
Enter the client's doctor's name:

Dr. A
Enter the client's doctor's phone number:

442-231-3232
Enter the team name associated with this client:

team2
```

```
2
Enter the client's social security number:

4
Enter the client's name
Y
Enter the client's gender:

M
Enter the client's profession:

Artist
Is the client on the mailing list? (Y or N)

Y
Enter the client's mailing address:

101 St
Enter the client's email address:

a@mail.com
Enter the client's phone number:

222-000-0000
Enter the client's assignment date (YYYY-MM-DD):

2021-01-22
Enter the client's doctor's name:

Dr. A
Enter the client's doctor's phone number:

121-121-4343
Enter the team name associated with this client:

team3
```

```
2
Enter the client's social security number:

5
Enter the client's name
O
Enter the client's gender:

F
Enter the client's profession:

doctor
Is the client on the mailing list? (Y or N)

y
Enter the client's mailing address:

11102 Rd
Enter the client's email address:

o@mail.com
Enter the client's phone number:

111-000-0000
Enter the client's assignment date (YYYY-MM-DD):

2010-01-01
Enter the client's doctor's name:

Dr. L
Enter the client's doctor's phone number:

2221111000
Enter the team name associated with this client:

team1
```

Result of queries:

Results Messages

	ssn	pname	gender	profession	on_mailing_list	mailing_addr	email_addr	phone_num	assignment_date	doctor_name	doctor_phone_number
1	1	A	F	Teacher	Y	a@gmail.com	111 Dr	444-000-11111	2021-02-02	Dr. J	333-000-0000
2	2	J	M	Salesman	N	111 Dr	j@gmail.com	444-000-1111	2022-02-02	Dr. J	333-000-0000
3	3	T	F	Doctor	Y	2324 Dr	t@mail.com	333-000-0000	2024-01-11	Dr. A	442-231-3232
4	4	Y	M	Artist	Y	101 St	a@mail.com	222-000-0000	2021-01-22	Dr. A	121-121-4343
5	5	O	F	doctor	y	11102 Rd	o@mail.com	111-000-0000	2010-01-01	Dr. L	2221111000

Task 6.3 Screenshots showing the testing of query 3

```
Enter the volunteer's social security number:

8
Enter the volunteer's name:

Vol1
Enter the volunteer's gender:

F
Enter the volunteer's profession:

teacher
Is the volunteer on the mailing list? (Y or N)

Y
Enter the volunteer's mailing address:

mail@mail.com
Enter the volunteer's email address:

mail@mail.com
Enter the volunteer's phone number:

111-000-0000
Enter the volunteer's date joined (YYYY-MM-DD):

2023-03-03
Enter the volunteer's date training (YYYY-MM-DD):

2024-03-03

Enter the training location:
training
Enter the team name associated with this volunteer:

team1
Enter the month the volunteer served:
```

```
Enter the month the volunteer served:

Aug
Enter the number of hours the volunteer served:

100
Indicate the status of the volunteer on the team:

active
```

```
3
Enter the volunteer's social security number:

9
Enter the volunteer's name:

Vol2
Enter the volunteer's gender:

M
Enter the volunteer's profession:

teacher
Is the volunteer on the mailing list? (Y or N)

N
Enter the volunteer's mailing address:

222 St
Enter the volunteer's email address:

vol@mail.com
Enter the volunteer's phone number:

111-111-0000
Enter the volunteer's date joined (YYYY-MM-DD):

2022-02-11
Enter the volunteer's date training (YYYY-MM-DD):

2023-03-11

Enter the training location:
training
Enter the team name associated with this volunteer:

team2
```

```
Enter the month the volunteer served:

Aug
Enter the number of hours the volunteer served:

100
Indicate the status of the volunteer on the team:

inactive
```

```
10
Enter the volunteer's name:

Vol3
Enter the volunteer's gender:

M
Enter the volunteer's profession:

artist
Is the volunteer on the mailing list? (Y or N)

N
Enter the volunteer's mailing address:

111 Dr
Enter the volunteer's email address:

mail@mail.com
Enter the volunteer's phone number:

111-1111-2222
Enter the volunteer's date joined (YYYY-MM-DD):

2024-04-04
Enter the volunteer's date training (YYYY-MM-DD):

2024-10-11

Enter the training location:
training
Enter the team name associated with this volunteer:

team3
```

```
Enter the month the volunteer served:

Aug
Enter the number of hours the volunteer served:

100
Indicate the status of the volunteer on the team:

active
```

```
Enter the volunteer's social security number:

11
Enter the volunteer's name:

Vol4
Enter the volunteer's gender:

M
Enter the volunteer's profession:

doctor
Is the volunteer on the mailing list? (Y or N)

Y
Enter the volunteer's mailing address:

222 Dr
Enter the volunteer's email address:

mail@mail.com
Enter the volunteer's phone number:

111-222-3333
Enter the volunteer's date joined (YYYY-MM-DD):

2024-04-01
Enter the volunteer's date training (YYYY-MM-DD):

2024-05-05

Enter the training location:
training
Enter the team name associated with this volunteer:

team1
```

```
Enter the month the volunteer served:

Dec
Enter the number of hours the volunteer served:

100
Indicate the status of the volunteer on the team:

active
```

```
PAN [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe  (Nov 10, 2024, 7:17:49 PM) [p
3
Enter the volunteer's social security number:

12
Enter the volunteer's name:

Vol5
Enter the volunteer's gender:

F
Enter the volunteer's profession:

photographer
Is the volunteer on the mailing list? (Y or N)

Y
Enter the volunteer's mailing address:

235 Dr
Enter the volunteer's email address:

mail@mail.com
Enter the volunteer's phone number:

111-222-3333
Enter the volunteer's date joined (YYYY-MM-DD):

2021-02-12
Enter the volunteer's date training (YYYY-MM-DD):

2022-01-01

Enter the training location:
training
Enter the team name associated with this volunteer:

team3
```

```
Enter the month the volunteer served:

Dec
Enter the number of hours the volunteer served:

20
Indicate the status of the volunteer on the team:

team5
```

Results for these queries:

* For ssn's 8, 9, 10, 11, 12. The top table is for ServesOn and the bottom table is Volunteers.

	ssn	team_name	serve_month	serve_hours	active
1	6	team1	Aug	100	
2	7	team2	AUg	1000	
3	6	team1	Aug	100	
4	8	team1	Aug	100	active
5	9	team2	Aug	100	inactive
6	9	team3	Nov	50	active
7	10	team3	Aug	100	active
8	11	team1	Dec	100	active
9	12	team3	Dec	20	team5

	ssn	pname	gender	profession	on_mailing_list	mailing_addr	email_addr	phone_num	date_joined	date_training	location_training
1	10	Vol3	M	artist	N	111 Dr	mail@mail.com	111-1111-2222	2024-04-04	2024-10-11	training
2	11	Vol4	M	doctor	Y	222 Dr	mail@mail.com	111-222-3333	2024-04-01	2024-05-05	training
3	12	Vol5	F	photographer	Y	235 Dr	mail@mail.com	111-222-3333	2021-02-12	2022-01-01	training
4	6	Vol1	F	doctor	Y	123 St	vol@mail.com	333-000-0000	2022-02-02	2023-11-11	location
5	7	Vol2	F	teacher	Y	555 St	email@mail.com	333-333-3333	2022-01-01	2022-03-03	training
6	8	Vol1	F	teacher	Y	mail@mail.com	mail@mail.com	111-000-0000	2023-03-03	2024-03-03	training
7	9	Vol2	M	teacher	N	222 St	vol@mail.com	111-111-0000	2022-02-11	2023-03-11	training

Task 6.4 Screenshots showing the testing of query 4

```
4
Enter the volunteer's social security number:

8
Enter the volunteer's team name:

team1
Enter the month the volunteer served:

Sept
Enter the hours the volunteer served:


100
Is the volunteer active?

active
```

```
4
Enter the volunteer's social security number:

9
Enter the volunteer's team name:

team2
Enter the month the volunteer served:

Sept
Enter the hours the volunteer served:

10
Is the volunteer active?

active
```

```
4
Enter the volunteer's social security number:

10
Enter the volunteer's team name:

team4
Enter the month the volunteer served:

Nov
Enter the hours the volunteer served:

15
Is the volunteer active?

active
```

```
4
Enter the volunteer's social security number:

11
Enter the volunteer's team name:

team4
Enter the month the volunteer served:

Dec
Enter the hours the volunteer served:

0
Is the volunteer active?

inactive
```

```
4
Enter the volunteer's social security number:

12
Enter the volunteer's team name:

team2
Enter the month the volunteer served:

Jan
Enter the hours the volunteer served:

5
Is the volunteer active?

active
```

Results for these queries:

Results | Messages

	ssn	team_name	serve_month	serve_hours	active
1	6	team1	Aug	100	
2	7	team2	AUg	1000	
3	6	team1	Aug	100	
4	8	team1	Aug	100	active
5	9	team2	Aug	100	inactive
6	9	team3	Nov	50	active
7	10	team3	Aug	100	active
8	11	team1	Dec	100	active
9	12	team3	Dec	20	team5
10	8	team1	Sept	100	active
11	9	team2	Sept	10	active
12	10	team4	Nov	15	active
13	11	team4	Dec	0	inactive
14	12	team2	Jan	5	active

Task 6.5 Screenshots showing the testing of query 5

```
Enter the employee's social security number:

13

Enter the employee's name:
Employee1

Enter the employee's gender:
F

Enter the employee's profession:
teacher

Is the employee on the mailing list? (Y or N):
Y

Enter the employee's mailing address:
1234 St

Enter the employee's email address:
mail@mail.com

Enter the employee's phone number:
111-111-0000
Enter the employee's salary:

10000

Enter the employee's marital status:
M

Enter the employee's hire date(YYYY-MM-DD):
2023-03-03

Enter the team name associated with this employee:
team1

Enter the report status:
done
```

```
Enter the report description:
asdksjdad

Enter date of the report (YYYY-MM-DD):
2023-03-03
```

```
Enter the employee's social security number:

14

Enter the employee's name:
Employee2

Enter the employee's gender:
F

Enter the employee's profession:
worker

Is the employee on the mailing list? (Y or N):
N

Enter the employee's mailing address:
12342 NW St

Enter the employee's email address:
mail@mail.com

Enter the employee's phone number:
222-000-1212
Enter the employee's salary:

23000

Enter the employee's marital status:
M

Enter the employee's hire date(YYYY-MM-DD):
2023-11-02

Enter the team name associated with this employee:
team2

Enter the report status:
in progress
```

```
Enter the report description:
sdkjsadjksa

Enter date of the report (YYYY-MM-DD):
2024-04-23
```

```
Enter the employee's social security number:

15

Enter the employee's name:
Employee4

Enter the employee's gender:
F

Enter the employee's profession:
doctor

Is the employee on the mailing list? (Y or N):
Y

Enter the employee's mailing address:
123 St

Enter the employee's email address:
mail@mail.com

Enter the employee's phone number:
405-000-0000
Enter the employee's salary:

120000

Enter the employee's marital status:
M

Enter the employee's hire date(YYYY-MM-DD):
2024-04-12

Enter the team name associated with this employee:
team2
```

```
Enter the report status:
done

Enter the report description:
skadjsakdj

Enter date of the report (YYYY-MM-DD):
2024-10-12
```

```
Enter the employee's social security number:

16

Enter the employee's name:
Employee5

Enter the employee's gender:
F

Enter the employee's profession:
teacher

Is the employee on the mailing list? (Y or N):
Y

Enter the employee's mailing address:
11123 St

Enter the employee's email address:
mail@mail.com

Enter the employee's phone number:
112-232-2222
Enter the employee's salary:

100000

Enter the employee's marital status:
M

Enter the employee's hire date(YYYY-MM-DD):
2024-01-01

Enter the team name associated with this employee:
team4
```

```
Enter the report status:
done

Enter the report description:
sadakdjsadas

Enter date of the report (YYYY-MM-DD):
2023-12-01
```

```
Enter the employee's social security number:

17

Enter the employee's name:
Employee3

Enter the employee's gender:
M

Enter the employee's profession:
teacher

Is the employee on the mailing list? (Y or N):
Y

Enter the employee's mailing address:
111 St

Enter the employee's email address:
mail@mail.com

Enter the employee's phone number:
111-000-2222
Enter the employee's salary:

134000

Enter the employee's marital status:
M

Enter the employee's hire date(YYYY-MM-DD):
2024-05-20

Enter the team name associated with this employee:
team4

Enter the report status:
Done
```

```
Enter the report description:
aksdsajda

Enter date of the report (YYYY-MM-DD):
2024-01-01
```

Results for these queries:

	ssn	pname	gender	profession	on_mailing_list	mailing_addr	email_addr	phone_num	salary	marital_status	hire_date
1	13	Employee1	F	teacher	Y	1234 St	mail@mail.com	111-111-0000	10000	M	2023-03-03
2	14	Employee2	F	worker	N	12342 NW St	mail@mail.com	222-000-1212	23000	M	2023-11-02
3	15	Employee4	F	doctor	Y	123 St	mail@mail.com	405-000-0000	120000	M	2024-04-12
4	16	Employee5	F	teacher	Y	11123 St	mail@mail.com	112-232-2222	100000	M	2024-01-01
5	17	Employee3	M	teacher	Y	111 St	mail@mail.com	111-000-2222	134000	M	2024-05-20

Task 6.6 Screenshots showing the testing of query 6

```
Enter the employee's ssn:
13

Etner the expense date:
2024-02-02

Enter the expense amount:
1000

Enter the expense description:
Supplies
```

```
Enter the employee's ssn:
14

Etner the expense date:
2024-11-01

Enter the expense amount:
2000

Enter the expense description:
Travel
```

```
Enter the employee's ssn:
15

Etner the expense date:
2020-12-02

Enter the expense amount:
100

Enter the expense description:
Food
```

```
Enter the employee's ssn:
15

Etner the expense date:
2020-01-01

Enter the expense amount:
10

Enter the expense description:
Supplies
```

```
Enter the employee's ssn:
16

Etner the expense date:
2024-11-10

Enter the expense amount:
200

Enter the expense description:
Furniture
```

Results for these queries:

	ssn	expense_date	expense_amount	expense_description
1	13	2024-02-02	1000.00	Supplies
2	14	2024-11-01	2000.00	Travel
3	15	2020-12-02	100.00	Food
4	15	2020-01-01	10.00	Supplies
5	16	2024-11-10	200.00	Furniture

Task 6.7 Screenshots showing the testing of query 7

```
Enter the donor's name:
Donor

Enter the donor's gender:
F

Enter the donor's profession:
Dentist

Is the donor on the mailing list? (Y or N):
Y

Enter the donor's mailing address:
111 St

Enter the donor's email address:
mail@mail.com

Enter the donor's phone number:
333-000-1111

Does the donor wish to remain anonymous? (Y or N):
Y

Enter date of the donation (YYYY-MM-DD):
2021-02-02

Enter donation amount:
2000

Enter the type of donation:
Check

Enter the campaign name:
C1
```

```
Enter the donor's social security number:

19

Enter the donor's name:
Donor2

Enter the donor's gender:
M

Enter the donor's profession:
teacher

Is the donor on the mailing list? (Y or N):
Y

Enter the donor's mailing address:
123 Dr

Enter the donor's email address:
mail@mail.com

Enter the donor's phone number:
111-222-0000

Does the donor wish to remain anonymous? (Y or N):
Y

Enter date of the donation (YYYY-MM-DD):
2021-01-01

Enter donation amount:
100

Enter the type of donation:
Check

Enter the campaign name:
C2
```

```
Enter the donor's social security number:

20

Enter the donor's name:
Donor3

Enter the donor's gender:
F

Enter the donor's profession:
Doctor

Is the donor on the mailing list? (Y or N):
Y

Enter the donor's mailing address:
12413 St

Enter the donor's email address:
mail@mail.com

Enter the donor's phone number:
000-111-2222

Does the donor wish to remain anonymous? (Y or N):
Y

Enter date of the donation (YYYY-MM-DD):
2022-01-01

Enter donation amount:
200

Enter the type of donation:
Card

Enter the campaign name:
C3
```

```
Enter the donor's social security number:

21

Enter the donor's name:
Donor4

Enter the donor's gender:
M

Enter the donor's profession:
Businessman

Is the donor on the mailing list? (Y or N):
N

Enter the donor's mailing address:
12321

Enter the donor's email address:
mail@mail.com

Enter the donor's phone number:
111-000-11111

Does the donor wish to remain anonymous? (Y or N):
N

Enter date of the donation (YYYY-MM-DD):
2021-02-02

Enter donation amount:
200000

Enter the type of donation:
Check

Enter the campaign name:
C1
```

```
Enter the donor's social security number:

22

Enter the donor's name:
Donor5

Enter the donor's gender:
M

Enter the donor's profession:
Doctor

Is the donor on the mailing list? (Y or N):
Y

Enter the donor's mailing address:
123 St

Enter the donor's email address:
mail@mail.com

Enter the donor's phone number:
222-000-1231

Does the donor wish to remain anonymous? (Y or N):
Y

Enter date of the donation (YYYY-MM-DD):
2021-01-01

Enter donation amount:
2000

Enter the type of donation:
Card

Enter the campaign name:
C3
```

Results for these queries:

	ssn	pname	gender	profession	on_mailing_list	mailing_addr	email_addr	phone_num	is_anonymous
1	18	Donor	F	Dentist	Y	111 St	mail@mail.com	333-000-1111	Y
2	19	Donor2	M	teacher	Y	123 Dr	mail@mail.com	111-222-0000	Y
3	20	Donor3	F	Doctor	Y	12413 St	mail@mail.com	000-111-2222	Y
4	21	Donor4	M	Businessman	N	12321	mail@mail.com	111-000-11111	N
5	22	Donor5	M	Doctor	Y	123 St	mail@mail.com	222-000-1231	Y

Task 6.8 Screenshots showing the testing of query 8

```
Enter the ssn of the client:
1
Doctor name:Dr. J
Doctor's phone number:333-000-0000
```

```
Enter the ssn of the client:
5
Doctor name:Dr. L
Doctor's phone number:2221111000
```

Task 6.9 Screenshots showing the testing of query 9

```
9
Enter the start time for expenses:
2000-01-01
Enter the end time for expenses:
2024-12-31
15: 110.0
16: 200.0
13: 1000.0
14: 2000.0
```

```
9
Enter the start time for expenses:
2000-01-01
Enter the end time for expenses:
2022-01-01
15: 110.0
```

Task 6.10 Screenshots showing the testing of query 10

```
Enter the ssn of the client:
1

Volunteers:
Vol1
Vol4
```

```
Enter the ssn of the client:
3

Volunteers:
Vol2
Vol5
```

Task 6.11 Screenshots showing the testing of query 11

```
11
Enter a start date
2000-01-01

Teams founded after 2000-01-01:
team1
team2
team3
team4
team5
```

```
Enter a start date
2022-01-01

Teams founded after 2022-01-01:
team2
team3
team5
```

Task 6.12 Screenshots showing the testing of query 12

*A, J, O, T, Y are the clients in the database

```
All people in the database:
Name       ssn        Mailing Address      Phone Number         Email Address        Emergency Contact Name
-----------------------------------------------------------------------------------------------------------------
A          1          a@gmail.com          444-000-11111        111 Dr               John
Donor      18         111 St               333-000-1111         mail@mail.com        E
Donor2     19         123 Dr               111-222-0000         mail@mail.com        G
Donor3     20         12413 St             000-111-2222         mail@mail.com        H
Donor4     21         12321                111-000-11111        mail@mail.com        I
Donor5     22         123 St               222-000-1231         mail@mail.com        I
Employee1  13         1234 St              111-111-0000         mail@mail.com        J
Employee2  14         12342 NW St          222-000-1212         mail@mail.com        K
Employee3  17         111 St               111-000-2222         mail@mail.com        N
Employee4  15         123 St               405-000-0000         mail@mail.com        L
Employee5  16         11123 St             112-232-2222         mail@mail.com        M
J          2          111 Dr               444-000-1111         j@gmail.com          A
O          5          11102 Rd             111-000-0000         o@mail.com           D
T          3          2324 Dr              333-000-0000         t@mail.com           B
Vol1       8          mail@mail.com        111-000-0000         mail@mail.com        O
Vol2       9          222 St               111-111-0000         vol@mail.com         P
Vol3       10         111 Dr               111-1111-2222        mail@mail.com        H
Vol4       11         222 Dr               111-222-3333         mail@mail.com        I
Vol5       12         235 Dr               111-222-3333         mail@mail.com        J
Y          4          101 St               222-000-0000         a@mail.com           C
```

Task 6.13 Screenshots showing the testing of query 13

For this query, I had to insert another donor on a person who is already an employee, so I choose ssn = 13. Here is the query for that:

```
7
Enter the donor's social security number:

13

Enter the donor's name:
DonorAndEmployee

Enter the donor's gender:
F

Enter the donor's profession:
Donor

Is the donor on the mailing list? (Y or N):
Y

Enter the donor's mailing address:
111

Enter the donor's email address:
111

Enter the donor's phone number:
111

Does the donor wish to remain anonymous? (Y or N):
N

Enter date of the donation (YYYY-MM-DD):
2020-01-01

Enter donation amount:
1000

Enter the type of donation:
check
```

Here is the main query for task 6.13:

```
13

Donations from donors that are also employees:
Donor Name           Total Amount
-----------------------------------------------------
DonorAndEmployee     1000
```

Task 6.14 Screenshots showing the testing of query 14

Message produced by the Java program:

```
14
Increasing salary by 1.1% ...
```

The original salaries:

	ssn	pname	gender	profession	on_mailing_list	mailing_addr	email_addr	phone_num	salary	m
1	13	Employee1	F	teacher	Y	1234 St	mail@mail.com	111-111-0000	10000	
2	14	Employee2	F	worker	N	12342 NW St	mail@mail.com	222-000-1212	23000	
3	15	Employee4	F	doctor	Y	123 St	mail@mail.com	405-000-0000	120000	
4	16	Employee5	F	teacher	Y	11123 St	mail@mail.com	112-232-2222	100000	
5	17	Employee3	M	teacher	Y	111 St	mail@mail.com	111-000-2222	134000	

The Reports table, showing which employees are reporting to teams:

	ssn	team_name	report_status	report_description	report_date
1	13	team1	done	asdksjdad	2023-03-03
2	14	team2	in progress	sdkjsadjksa	2024-04-23
3	14	team2	done	djsajasdas	2024-10-10
4	15	team2	done	skadjsakdj	2024-10-12
5	16	team4	done	sadakdjsadas	2023-12-01
6	17	team4	Done	aksdsajda	2024-01-01

The new salaries:

	ssn	pname	gender	profession	on_mailing_list	mailing_addr	email_addr	phone_num	salary
1	13	Employee1	F	teacher	Y	1234 St	mail@mail.com	111-111-0000	10000
2	14	Employee2	F	worker	N	12342 NW St	mail@mail.com	222-000-1212	25300
3	15	Employee4	F	doctor	Y	123 St	mail@mail.com	405-000-0000	120000
4	16	Employee5	F	teacher	Y	11123 St	mail@mail.com	112-232-2222	100000
5	17	Employee3	M	teacher	Y	111 St	mail@mail.com	111-000-2222	134000

The results of this query are that the employee whose SSN is 14 increased their salary by 1.1%, from $23,000 to $25,300.

Task 6.15 Screenshots showing the testing of query 15

Message produced by the Java program:

```
15
Deleting clients without insurance ...
```

Here are the Needs:

	ssn	need	importance_value
1	1	Transportation	1
2	2	Other	10
3	3	Transportation	1
4	4	Health	5
5	5	Transportation	10

Here are the Insurance Policies:

	policy_ID	provider_name	provider_addr	insurance_type
1	1	a	a	NotHealth
2	2	a	a	Health
3	3	a	a	NotHealth
4	4	a	a	Health
5	5	a	a	Health

Here are the clients before being deleted:

Results Messages

	ssn	pname	gender	profession	on_mailing_list	mailing_addr	email_addr	phone_num	assignment_date	doctor_name	doctor_phone_number
1	1	A	F	Teacher	Y	a@gmail.com	111 Dr	444-000-11111	2021-02-02	Dr. J	333-000-0000
2	2	J	M	Salesman	N	111 Dr	j@gmail.com	444-000-1111	2022-02-02	Dr. J	333-000-0000
3	3	T	F	Doctor	Y	2324 Dr	t@mail.com	333-000-0000	2024-01-11	Dr. A	442-231-3232
4	4	Y	M	Artist	Y	101 St	a@mail.com	222-000-0000	2021-01-22	Dr. A	121-121-4343
5	5	O	F	doctor	y	11102 Rd	o@mail.com	111-000-0000	2010-01-01	Dr. L	2221111000

Here are the results after:

	ssn	pname	gender	profession	on_mailing_list	mailing_addr	email_addr	phone_num	assignment_date	doctor_name	doctor_phone_number
1	2	J	M	Salesman	N	111 Dr	j@gmail.com	444-000-1111	2022-02-02	Dr. J	333-000-0000
2	4	Y	M	Artist	Y	101 St	a@mail.com	222-000-0000	2021-01-22	Dr. A	121-121-4343
3	5	O	F	doctor	y	11102 Rd	o@mail.com	111-000-0000	2010-01-01	Dr. L	2221111000

Task 6.16. Screenshots showing the testing of the import and export options

Import:

Data from the importing file (test.csv)

team6	Health	2020-01-01
team7	Lung	2023-11-01
team8	Heart	2024-01-01

Running the query:

```
16

Enter the file name to load data from:
test.csv
```

Results of the Teams table:

	team_name	team_type	date_formed
1	team1	Health	2022-01-01
2	team2	Heart	2023-02-02
3	team3	Lung	2023-11-02
4	team4	Skin	2021-01-01
5	team5	Eyes	2023-03-01
6	team6	Health	2020-01-01
7	team7	Lung	2023-11-01
8	team8	Heart	2024-01-01

Export:
Running the query:

```
17

Enter a file name to output to:
out.csv
```

Result of export (out.csv):

A	a@gmail.com	
Donor	111 St	
Donor2	123 Dr	
Donor3	12413 St	
Donor5	123 St	
DonorAndE	111	
Employee1	1234 St	
Employee3	111 St	
Employee4	123 St	
Employee5	11123 St	
O	11102 Rd	
T	2324 Dr	
Vol1	123 St	
Vol1	mail@mail.com	
Vol2	555 St	
Vol4	222 Dr	
Vol5	235 Dr	
Y	101 St	

Task 6.17. Screenshots showing the testing of three types of errors

Error 1 – Illegal Argument Exception

```
Enter the date that the team formed (YYYY-MM-DD):

1
Exception in thread "main" java.lang.IllegalArgumentException
        at java.sql/java.sql.Date.valueOf(Date.java:141)
        at PAN.enterTeam(PAN.java:178)
        at PAN.main(PAN.java:83)
```

Error 2 – Foreign Key constraint

```
INSERT INTO Needs
VALUES('1', 'Transportation', 1),
('2', 'Other', 10),
('3', 'Transportation', 1),
('4', 'Health', 5),
('5', 'Transportation', 10);
```

Message:

```
Started executing query at Line 25
Msg 547, Level 16, State 0, Line 1
The INSERT statement conflicted with the FOREIGN KEY constraint "FK__Needs__ssn__5689C04F". The conflict occurred in database "cs-dsa-4513-sql-db", table "dbo.Clients", column 'ssn'.
The statement has been terminated.
Total execution time: 00:00:00.046
```

Error 3 – File not found error

```
Enter the file name to load data from:
file.csv
Exception in thread "main" java.io.FileNotFoundException: src\file.csv (The system cannot find the file specified)
        at java.base/java.io.FileInputStream.open0(Native Method)
        at java.base/java.io.FileInputStream.open(FileInputStream.java:213)
        at java.base/java.io.FileInputStream.<init>(FileInputStream.java:152)
        at java.base/java.io.FileInputStream.<init>(FileInputStream.java:106)
        at java.base/java.io.FileReader.<init>(FileReader.java:60)
        at PAN.importData(PAN.java:820)
        at PAN.main(PAN.java:143)
```

Task 6.18. Screenshots showing the testing of the quit option

```
PAN.java   example.java
751                     rs.getString("pname"), rs.getString("ssn"), rs.getString("mailing_addr"),
752                     rs.getString("phone_num"), rs.getString("email_addr"),

Console
<terminated> PAN [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe  (Nov 11, 2024, 12:15:54 AM – 12:33:56 AM) [pid: 26708]
9) Retrieve the total amount of expenses charged by each employee for a particular period of time. The list should be sorte
10) Retrieve the list of volunteers that are members of teams that support a particular client
11) Retrieve the names of all teams that were founded after a particular date
12) Retrieve the names, social security numbers, contact information, and emergency contact information of all people in th
13) Retrieve the name and total amount donated by donors that are also employees. The list should be sorted by the total am
14) Increase the salary by 10% of all employees to whom more than one team must report
15) Delete all clients who do not have health insurance and whose value of importance for transportation is less than 5
16) Import: enter new teams from a data file until the file is empty (the user must be asked to enter the input file name).
17) Export: Retrieve names and mailing addresses of all people on the mailing list and output them to a data file instead o
18) Quit

17

Enter a file name to output to:
out.csv

Please select one of the options below:
1) Enter a new team into the database
2) Enter a new client into the database and associate him or her with one or more teams
3) Enter a new volunteer into the database and associate him or her with one or more teams
4) Enter the number of hours a volunteer worked this month for a particular team
5) Enter a new employee into the database and associate him or her with one or more teams
6) Enter an expense charged by an employee
7) Enter a new donor and associate him or her with several donations
8) Retrieve the name and phone number of the doctor of a particular client
9) Retrieve the total amount of expenses charged by each employee for a particular period of time. The list should be sorte
10) Retrieve the list of volunteers that are members of teams that support a particular client
11) Retrieve the names of all teams that were founded after a particular date
12) Retrieve the names, social security numbers, contact information, and emergency contact information of all people in th
13) Retrieve the name and total amount donated by donors that are also employees. The list should be sorted by the total am
14) Increase the salary by 10% of all employees to whom more than one team must report
15) Delete all clients who do not have health insurance and whose value of importance for transportation is less than 5
16) Import: enter new teams from a data file until the file is empty (the user must be asked to enter the input file name).
17) Export: Retrieve names and mailing addresses of all people on the mailing list and output them to a data file instead o
18) Quit

18
Closing the PAN Database system...
```