

## **A PATIENT ASSISTANT NETWORK DATABASE SYSTEM**

**Name: Meghnath Reddy Challa**

**OU ID :113508098**

**Email: [challa.reddy01@ou.edu](mailto:challa.reddy01@ou.edu)**

**Course name: DataBase Management Systems**

**Course number:4513-001**

**Section: DSA-001**

**Year:2020 Fall Semester**

**Instructor Name: Dr. Le Gruenwald**

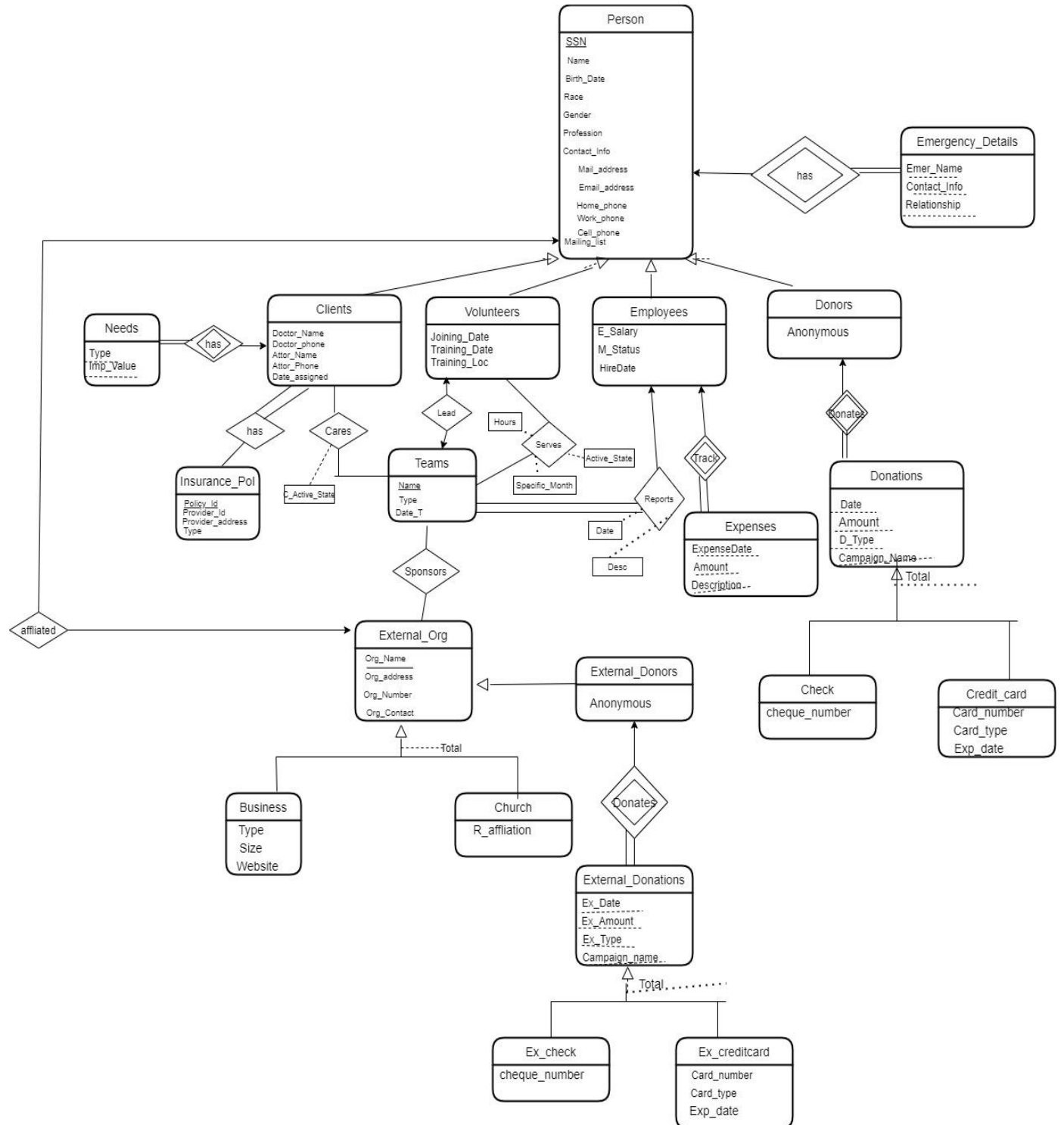
## Contents

TASK-1.....	4
1.1 ER Diagram .....	4
1.2. Relational Database Schema .....	5
Task 2 .....	6
2.1 DATA DICTIONARY.....	6
TASK 3 .....	13
3.1 Discussion of storage structures for tables.....	13
3.2 Discussion of storage structures for tables (Azure SQL Database) .....	15
TASK 4: SQL statements and screenshots showing the creation of tables in Azure SQL Database.....	17
Task 5 .....	27
5.1 SQL statements and Transact SQL stored procedures .....	27
5.2 The Java source program and screenshots showing its successful compilation .....	33
TASK 6 .....	50
6.1) Screenshot showing successful execution of option 1.....	50
6.2) Screenshot showing successful execution of option 2.....	54
6.3) Screenshot showing successful execution of option 3.....	58
6.4) Screenshot showing successful execution of option 4.....	61
6.5) Screenshot showing successful execution of option 5.....	63
6.6) Screenshot showing successful execution of option 6.....	66
6.7) Screenshot showing successful execution of option 7.....	67
6.8) Screenshot showing successful execution of option 8.....	70
6.9) Screenshot showing successful execution of option 9.....	73
6.10) Screenshot showing successful execution of option 10.....	78
6.11) Screenshot showing successful execution of option 11.....	79
6.12) Screenshot showing successful execution of option 12.....	79
6.13) Screenshot showing successful execution of option 13.....	80
6.14) Screenshot showing successful execution of option 14.....	80
6.15) Screenshot showing successful execution of option 15.....	80
6.16) Screenshot showing successful execution of option 16.....	80

6.17) Screenshot showing successful execution of option 17 .....	80
6.18) Screenshot showing successful execution of option 18 .....	81
6.19) Screenshot showing successful execution of option 19 .....	82
6.20) Screenshot showing successful execution of option 20 .....	83
6.21) Screenshots to demonstrate Error handling .....	83
TASK 7 .....	85
7.1. Web database application source program and screenshots showing Its successful compilation .	85
7.1.1 Data handlers file .....	85
7.1.2 For Query 15 form file: .....	86
7.1.3 For Query 15 Teams file takes input from input form: .....	87
7.1.4 For Query 1 form file: .....	88
7.1.5 For Query 1 Teams file takes input from input form: .....	89
7.2. Screenshots showing the testing of the Web database application .....	90
7.3) Screenshots to demonstrate Query 1 works in JSP .....	91
7.4) Screenshots to demonstrate Query 15 works after updating Teams .....	92
7.5) Screenshots to demonstrate get_Teams function works.....	93

# TASK-1

## 1.1 ER Diagram



## 1.2. Relational Database Schema

Person (SSN, Name, BirthDate, Race, Gender, Profession, Mail\_address, Email\_address, Home\_phone, Work\_phone, Cell\_phone, Mailing\_list)

Clients (SSN, Doctor\_Name, Doctor\_phone, Attor\_Name, Attor\_Phone, Date\_assigned)

Volunteers (SSN, Joining\_Date, Training\_Date, Training\_Loc)

Employees(SSN, E\_Salary, M\_Status, HireDate)

Donors(SSN, Anonymous)

Emergency\_Details( SSN, Emer\_Name, Contact\_Info, Relationship)

Needs(SSN, Type, Imp\_Value)

Insurance\_Policy (SSN, Policy\_id, Provider\_Id, Provider\_address, Type)

Teams(T\_Name, Type, Date\_T)

Cares(SSN, T\_Name, C\_Active\_State)

Serves (SSN, T\_Name, Hours, Active\_State, Specific\_Month)

Reports(T\_Name, SSN, Date, Desc)

Expenses (SSN, ExpenseDate, Amount, Description)

Check(SSN, C\_Date, Amount, D\_Type, Campaign\_Name, cheque\_number)

Credit\_card(SSN, CC\_Date, CC\_Amount, D\_Type, Campaign\_Name, Card\_number, Card\_type, Exp\_Date)

Sponsors(T\_Name, Org\_Name)

Affiliated(SSN, Org\_Name)

External\_Org(Org\_Name, Org\_address, Org\_Number, Org\_Contact)

Business(Org\_Name, Type, Size, Website)

Church(Org\_Name, R\_affiliation)

External\_Donors(Org\_Name, Anonymous)

Ex\_check(Org\_Name, Ex\_Date, Ex\_Amount, Ex\_Type, Campaign\_name, cheque\_number)

Ex\_creditcard(Org\_Name, ExCC\_Date, ExCC\_Amount, ExCC\_Type, ExCCCampaign\_name, ExCard\_number, ExCard\_type, Ex\_Exp\_Date)

## Task 2

### 2.1 DATA DICTIONARY

**Person Table**

Attribute Name	Type	Size	Constraint
SSN	INT		Primary key
Name	Varchar	30	NOT NULL
BirthDate	Date		NOT NULL
Race	Varchar	20	NOT NULL
Gender	Varchar	10	NOT NULL
Profession	Varchar	20	NOT NULL
Mail_address	Varchar	100	NOT NULL
Email_address	Varchar	30	NOT NULL
Home_phone	INT		NOT NULL
Work_phone	INT		NOT NULL
Cell_phone	INT		NOT NULL
Mailing_list	INT		NOT NULL

**Clients Table**

Attribute Name	Type	Size	Constraint
SSN	INT		Primary key, Foreign key reference (SSN) from Person Table
Doctor_Name	Varchar	30	NOT NULL
Doctor_phone	INT		NOT NULL
Attor_Name	Varchar	30	NOT NULL
Attor_phone	INT		NOT NULL
Date_assigned	Date		NOT NULL

### Volunteers Table

Attribute Name	Type	Size	Constraint
SSN	INT		Primary key, Foreign key reference (SSN) from Person Table
Joining_Date	Date		NOT NULL
Training_Date	Date		NOT NULL
Training_Loc	Varchar	30	NOT NULL

### Employees Table

Attribute Name	Type	Size	Constraint
SSN	INT		Primary key, Foreign key reference (SSN) from Person Table
E_Salary	FLOAT		NOT NULL
M_Status	Varchar	20	NOT NULL
HireDate	Date		NOT NULL

### Donors Table

Attribute Name	Type	Size	Constraint
SSN	INT		Primary key, Foreign key reference (SSN) from Person Table
Anonymous	Varchar	30	NOT NULL

### Emergency Details Table

Attribute Name	Type	Size	Constraint
SSN	INT		Foreign key reference (SSN) from Person Table
Emer_Name	Varchar	30	NOT NULL
Contact_Info	INT		NOT NULL
Relationship	Varchar	30	NOT NULL

For Employee Details Table the primary key is formed by combination of  
( SSN, Emer\_Name, Contact\_Info, Relationship)

### Needs Table

Attribute Name	Type	Size	Constraint
SSN	INT		Foreign key reference (SSN) from Person Table
Type	Varchar	30	NOT NULL
Imp_Value	INT		NOT NULL, Imp_Value between 1 to 10.

For **Needs Table**: Primary key ( SSN,Type,Imp\_Value)

### Insurance Policy Table

Attribute Name	Type	Size	Constraint
SSN	INT		Foreign key reference (SSN) from Person Table
Policy_id	INT		Primary key
Provider_id	INT		Not Null
Provider_address	Varchar	100	Not null
Type	Varchar	30	Not null

### Teams Table

Attribute Name	Type	Size	Constraint
T_Name	Varchar	30	Primary key
Type	Varchar	20	Not Null
Date_T	INT		Not Null

### Cares Table

Attribute Name	Type	Size	Constraint
SSN	INT		Foreign key reference (SSN) from Person Table
T_Name	Varchar	30	Foreign key reference (T_Name) from Teams Table
C_ActiveState	INT		Not Null

For Cares Table primary key (SSN,T\_Name)



### Serves Table

Attribute Name	Type	Size	Constraint
SSN	INT		Foreign key reference (SSN) from Person Table
T_Name	Varchar	30	Foreign key reference (T_Name) from Teams Table
Hours	INT		Not Null
Active_State	INT		Not Null
Specific_Month	Varchar	20	Not Null

For Serves Table primary key: SSN,T\_Name

### Reports Table

Attribute Name	Type	Size	Constraint
SSN	INT		Foreign key reference (SSN) from Person Table
T_Name	Varchar	30	Foreign key reference (T_Name) from Teams Table
R_Date	Date		Not Null
Desc	Varchar	30	Not Null

For Reports Primary key: SSN,T\_Name

### Expenses Table

Attribute Name	Type	Size	Constraint
SSN	INT		Primary key, Foreign key reference (SSN) from Person Table
Amount	INT		Not Null
Expenses_Date	INT		Not Null
Desc	Varchar	30	Not Null

### Check\_1 Table

Attribute Name	Type	Size	Constraint
SSN	INT		Foreign key reference (SSN) from Person Table
C_Date	Date		NOT NULL
Amount	INT		NOT NULL
D_Type	Varchar	30	NOT NULL
Campaign_Name	Varchar	30	NOT NULL
Cheque_Number	INT		NOT NULL

For Check\_1 Table Primary key (SSN, C\_Date, Amount, D\_Type, Campaign\_Name)

### Credit Card Table

Attribute Name	Type	Size	Constraint
SSN	INT		Foreign key reference (SSN) from Person Table
CC_Date	Date		NOT NULL
CC_Amount	INT		NOT NULL
D_Type	Varchar	30	NOT NULL
Campaign_Name	Varchar	30	NOT NULL
Card_number	INT		NOT NULL
Card_Type	Varchar	20	NOT NULL
Exp_Date	Date		Not NULL

For Credit Card Table Primary key (SSN, CC\_Date, CC\_Amount, D\_Type, Campaign\_Name,)

### Sponsors Table

Attribute Name	Type	Size	Constraint
T_Name	Varchar	30	Foreign key reference (T_Name) from Teams Table
Org_Name	Varchar	20	Foreign key reference (Org_Name) from External_Org Table

For Sponsors Table Primary key : (T\_Name, Org\_Name)

### Affiliated Table

Attribute Name	Type	Size	Constraint
SSN	INT		Primary key, Foreign key reference (SSN) from Person Table
Org_Name	Varchar	20	Foreign key reference (Org_Name) from External_Org Table

### External\_Org Table

Attribute Name	Type	Size	Constraint
Org_Name	Varchar	20	Primary key
Org_address	Varchar	30	NOT NULL
Org_Number	INT		NOT NULL
Org_Contact	Varchar	20	NOT NULL

### Business Table

Attribute Name	Type	Size	Constraint
Org_Name	Varchar	20	Primary key, Foreign key reference (Org_Name) from External_Org Table
Type	Varchar	30	NOT NULL
Size	INT		NOT NULL
Website	Varchar	50	NOT NULL

### Church Table

Attribute Name	Type	Size	Constraint
Org_Name	Varchar	20	Primary key, Foreign key reference (Org_Name) from External_Org Table
R_affiliation	Varchar	20	NOT NULL

### External\_Donor

Attribute Name	Type	Size	Constraint
Org_Name	Varchar	20	Primary key, Foreign key reference (Org_Name) from External_Org Table
Anonymous	Varchar	20	NOT NULL

### Ex\_check Table

Attribute Name	Type	Size	Constraint
Org_Name	Varchar	20	Foreign key reference (Org_Name) from External_Org Table
Ex_Date	Date		NOT NULL
Ex_Amount	INT		NOT NULL
Ex_Type	Varchar	30	NOT NULL
Campaign_Name	Varchar	30	NOT NULL
Cheque_Number	INT		NOT NULL

For Ex\_Check Primary key: (Org\_Name, Ex\_Date, Ex\_Amount, Ex\_Type, Campaign\_name)

### Ex\_creditcard Table

Attribute Name	Type	Size	Constraint
Org_Name	Varchar	20	Foreign key reference (Org_Name) from External_Org Table
ExCC_Date	Date		NOT NULL
ExCC_Amount	INT		NOT NULL
ExCC_Type	Varchar	30	NOT NULL
ExCCCampaign_Name	Varchar	30	NOT NULL
ExCard_number	INT		NOT NULL
ExCard_Type	Varchar	20	NOT NULL
Ex_Exp_Date	Date		NOT NULL

For Ex\_Credit Table Primary key: (Org\_Name, ExCC\_Date, ExCC\_Amount, ExCC\_Type, ExCCCampaign\_name)

## TASK 3

### 3.1 Discussion of storage structures for tables

Table Name	Query# and Type	Search key	Query Frequency	Selected File Organization	Justification
<b>Person</b>	13.Random Search 14.Random Search	SSN SSN	1/week 1/week	Extendable (dynamic) hashing with SSN as hash key.	Good for random Search and no effort for searching and sorting required.
<b>Clients</b>	2.Insert 10.Random Search 17.Delete	SSN SSN	1/week 1/week 4/year	Extendable hashing with SSN as hash key.	Good for random Search and no effort for searching and sorting required.
<b>Volunteers</b>	3.Insert 4.Insert 12.Random Search	SSN	2/month 30/month 4/year	Index-Sequential	Since the indexing done by SSN will retrieve faster.
<b>Employees</b>	5.Insert 14.Random Search 6.Insert 11.Random Search 16.Update	SSN  SSN SSN	1/year 1/week 1/day 1/month 1/year	Extendable hashing with SSN as hash key.	Good for random Search and no effort for searching and sorting required.
<b>Donors</b>	8.Insert		1/day	Heap file.	Only insertion operation is to be performed.
<b>Emergency_details</b>				Heap file.	Unclear what operations will be performed on this table so using heap file.
<b>Needs</b>	17.Random Search	Type Imp_Value	4/year	Extendable hashing with Type as primary and Imp_Value as secondary index.	Good for random Search and no effort for searching and sorting required.
<b>Insurance_Policy</b>	17.Random Search	Type	4/year	Index sequential file with Policy_id as primary and Type as secondary index.	Good for random Search and no effort for searching and sorting required.
<b>Teams</b>	1. Insert 15.Random Search 2.Insert 3.Insert 4.Insert 5.Insert 7.Insert 12.Random Search 13. Range Search	Date_T      T_Name T_Name	1/month 1/month 1/week 2/month 30/month 1/year 2/week 4/year 1/week	B+ Tree	Since the records are stored in leaf and sorted in sequential linked list so search is fast
<b>Cares</b>	2.Insert 13.Random Search 12.Random Search	T_Name SSN	1/week 1/week 4/YEAR	Extendable hashing with SSN as hash key and T_name as	Good for random Search and no effort for searching and

				secondary index.	sorting required.
<b>Serves</b>	4.Insert 3.Insert 12.Random Search	T_Name	30/month 2/month 4/year	Index-Sequential	T_Name is used as Random Search.
<b>Reports</b>	5.Insert 16. Random Search	SSN	1/year 1/year	Heap file.	Only insertion operation is to be performed.
<b>Expenses</b>	6.Insert 11.Range Search	ExpenseDate	1/day 1/month	B+ tree with ExpenseDate as search key.	Since the records are stored in leaf and sorted in sequential linked list so search is fast.
<b>Check</b>	8.Insert		1/day	Heap file.	Only insertion operation is to be performed.
<b>Credit_Card</b>	8.Insert		1/day	Heap file.	Only insertion operation is to be performed.
<b>Sponsors</b>	7.Insert 13.Range Search	Org_Name	2/week 1/week	B+ tree with Org_Name as search key.	Since the records are stored in leaf and sorted in sequential linked list so search is fast.
<b>Affiliated</b>				Heap file.	Unclear what operations will be performed on this table so using heap file.
<b>External_Org</b>	7.Insert 9.Insert		2/week 1/day	Heap file.	Only insertion operation is to be performed.
<b>Business</b>				Heap file.	Unclear what operations will be performed on this table so using heap file.
<b>Church</b>				Heap file.	Unclear what operations will be performed on this table so using heap file.
<b>External_Donors</b>	9.Insert		1/day	Heap file.	Only insertion operation is to be performed.
<b>Ex_check</b>	9.Insert		1/day	Heap file.	Only insertion operation is to be performed.
<b>Ex_creditcard</b>	9.Insert		1/day	Heap file.	Only insertion operation is to be performed.

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

**Storage structure in Azure SQL DB** : It does not require us to implicitly mention on what attribute the indexing should be done if the attribute you want to index is already a primary key. As part of selecting the best file organization the decision would be taken by Azure SQL. The following changes would be as a result:

Table Name	Query# and Type	Search key	Query Frequency	Selected File Organization/Index requirement	Justification
Person	13.Random Search 14.Random Search	SSN SSN	1/week 1/week	No indexing required.	As SSN is already a primary key indexing would be done by default.
Clients	2.Insert 10.Random Search 17.Delete	SSN SSN	1/week 1/week 4/year	No indexing required.	As SSN is already a primary key indexing would be done by default.
Volunteers	3.Insert 4.Insert 12.Random Search	SSN	2/month 30/month 4/year	No indexing required.	Only insertion operation is to be performed.
Employees	5.Insert 14.Random Search 6.Insert 11.Random Search 16.Update	SSN SSN SSN	1/year 1/week 1/day 1/month 1/year	No indexing required.	As SSN is already a primary key indexing would be done by default.
Donors	8.Insert		1/day	No indexing required.	Only insertion operation is to be performed.
Emergency_details				No indexing required.	Unclear what operations will be performed on this table so using heap file.
Needs	17.Random Search	Type Imp_Value	4/year	Indexing required	Search is on Type and Value which is extra.
Insurance_Policy	17.Random Search	Type	4/year	Indexing required	Search is on Type.
Teams	1. Insert 15.Random Search 2.Insert 3.Insert 4.Insert 5.Insert 7.Insert 12.Random Search 13. Range Search	Date_T       T_Name T_Name	1/month 1/month 1/week 2/month 30/month 1/year 2/week 4/year 1/week	Indexing required	Search is on Date and Type.
Cares	2.Insert 13.Random Search 12.Random Search	T_Name SSN	1/week 1/week 4/YEAR	Indexing required	Search is on T_Name and SSN.
Serves	4.Insert 3.Insert 12.Random Search	T_Name	30/month 2/month 4/year	Indexing required	Search is on T_Name
Reports	5.Insert		1/year	No indexing	Only insertion

	16. Random Search	SSN	1/year	required.	operation is to be performed.
<b>Expenses</b>	6.Insert 11.Range Search	ExpenseDate	1/day 1/month	Indexing required	Search is on ExpenseDate
<b>Check</b>	8.Insert		1/day	No indexing required.	Only insertion operation is to be performed.
<b>Credit_Card</b>	8.Insert		1/day	No indexing required.	Only insertion operation is to be performed.
<b>Sponsors</b>	7.Insert 13.Range Search	Org_Name	2/week 1/week	Indexing required	Search is on Org_Name.
<b>Affiliated</b>				No indexing required.	Unclear what operations will be performed on this table so using heap file.
<b>External_Org</b>	7.Insert 9.Insert		2/week 1/day	No indexing required.	Only insertion operation is to be performed.
<b>Business</b>				No indexing required.	Unclear what operations will be performed on this table so using heap file.
<b>Church</b>				No indexing required.	Unclear what operations will be performed on this table so using heap file.
<b>External_Donors</b>	9.Insert		1/day	No indexing required.	Only insertion operation is to be performed.
<b>Ex_check</b>	9.Insert		1/day	No indexing required.	Only insertion operation is to be performed.
<b>Ex_creditcard</b>	9.Insert		1/day	No indexing required.	Only insertion operation is to be performed.



## TASK 4: SQL statements and screenshots showing the creation of tables in Azure SQL Database

-----CREATE TABLES-----

### 1) Person Table

```
CREATE Table Person (SSN INT PRIMARY KEY, Name Varchar(30) NOT NULL, Birthdate DATE NOT NULL, Race Varchar(20) NOT NULL, Gender Varchar(10) NOT NULL, Profession Varchar(20) NOT NULL, Mail_address Varchar(100), Email_address Varchar(30), Home_phone INT NOT NULL, Work_phone INT NOT NULL, Cell_phone INT NOT NULL, Mailing_list INT NOT NULL);
```

### 2) External\_Org Table

```
CREATE TABLE External_Org (Org_Name Varchar (20) PRIMARY KEY, Org_address Varchar (30) NOT NULL, Org_Number INT NOT NULL, Org_Contact Varchar(20) NOT NULL);
```

### 3) Client Table

```
CREATE TABLE Client (SSN INT PRIMARY KEY, Doctor_Name Varchar(30) NOT NULL, Attor_Name Varchar(30) NOT NULL, Doctor_phone INT NOT NULL, Attor_phone INT NOT NULL, Date_assigned DATE NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN) );
```

### 4) Employees Table

```
CREATE TABLE Employees (SSN INT PRIMARY KEY, E_Salary FLOAT NOT NULL, M_Status Varchar(20), HireDate DATE NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN) );
```

### 5) Volunteers Table

```
CREATE TABLE Volunteers (SSN INT PRIMARY KEY, Joining_Date DATE NOT NULL, Training_Date DATE NOT NULL, Training_Loc Varchar(30) NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN));
```

### 6) Donors Table

```
CREATE TABLE Donors (SSN INT PRIMARY KEY, Anonymous Varchar(30) NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN) );
```

### 7) Emergency\_Details Table

```
CREATE TABLE Emergency_Details (SSN INT, Emer_Name Varchar(30), Contact_Info INT, Relationship Varchar(30) ,PRIMARY KEY (Emer_Name, Contact_Info, Relationship), FOREIGN KEY (SSN) REFERENCES PERSON(SSN) );
```

#### 8) Needs Table

```
CREATE TABLE Needs (SSN INT, Type Varchar(30) , Imp_Value INT NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN),CONSTRAINT Value_check CHECK (Imp_Value between 1 and 10),PRIMARY KEY (SSN,Type, Imp_Value) );
```

#### 9) Insurance\_Policy Table

```
CREATE TABLE Insurance_Policy (SSN INT NOT NULL, Policy_id INT PRIMARY KEY, Provider_id INT NOT NULL, Provider_address Varchar(100) NOT NULL, Type VARCHAR(30) NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN));
```

#### 10) Teams Table

```
CREATE TABLE Teams ( T_Name Varchar(30) PRIMARY KEY, Type Varchar(20) NOT NULL, Date_T INT NOT NULL);
```

#### 11) Cares Table

```
CREATE TABLE Cares (SSN INT, T_Name Varchar(30),C_ActiveState INT NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN) , FOREIGN KEY (T_Name) REFERENCES Teams(T_Name) );
```

#### 12) Serves Table

```
CREATE TABLE Serves (SSN INT, FOREIGN KEY (SSN) REFERENCES PERSON(SSN), T_Name Varchar(30),FOREIGN KEY (T_Name) REFERENCES Teams(T_Name) ,Hours INT NOT NULL, Active_State INT NOT NULL, Specific_Month Varchar (20) NOT NULL);
```

#### 13) Reports Table

```
CREATE TABLE Reports (SSN INT, T_Name Varchar(30), R_DATE DATE NOT NULL, Desc_1 Varchar(30) NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN), FOREIGN KEY (T_Name) REFERENCES Teams(T_Name), PRIMARY KEY (T_Name,SSN) );
```

#### 14) Expenses Table

```
CREATE TABLE Expenses (SSN INT , FOREIGN KEY (SSN) REFERENCES PERSON(SSN),Amount FLOAT NOT NULL, Desc_1 Varchar (30) NOT NULL,Expense_Date INT NOT NULL,PRIMARY KEY(Amount,Desc_1,Expense_Date,SSN));
```

#### 15) Check\_1Table

```
CREATE TABLE Check_1(SSN INT, FOREIGN KEY (SSN) REFERENCES PERSON(SSN), C_Date DATE , Amount FLOAT , D_Type Varchar(30), Campaign_Name Varchar(30),Cheque_Number INT NOT NULL,PRIMARY KEY (SSN, C_Date, Amount, D_Type, Campaign_Name, Cheque_Number));
```

#### **16) Credit\_Card Table**

```
CREATE TABLE Credit_Card(SSN INT, FOREIGN KEY(SSN) REFERENCES PERSON(SSN),  
CC_Date DATE, CC_Amount FLOAT, D_Type Varchar(30), Campaign_Name Varchar(30), Card_Number  
INT NOT NULL, Card_Type Varchar(20) NOT NULL, Exp_Date DATE NOT NULL, PRIMARY KEY(SSN,  
CC_Date, CC_Amount, D_Type, Campaign_Name));
```

#### **17) Sponsors**

```
CREATE TABLE Sponsors(T_Name Varchar(30), FOREIGN KEY(T_Name) REFERENCES Teams(T_Name),  
Org_Name Varchar(20), FOREIGN KEY(Org_Name) REFERENCES External_Org(Org_Name));
```

#### **18) Affiliated Table**

```
CREATE TABLE Affiliated(SSN INT, FOREIGN KEY(SSN) REFERENCES PERSON(SSN), Org_Name Varchar  
(20), FOREIGN KEY(Org_Name) REFERENCES External_Org(Org_Name), PRIMARY KEY(SSN));
```

#### **19) Business Table**

```
CREATE TABLE Business(Org_Name Varchar(20) PRIMARY KEY, FOREIGN KEY(Org_Name) REFERENCES  
External_Org(Org_Name), Type Varchar(30) NOT NULL, Size INT NOT NULL, Website Varchar(50) NOT  
NULL);
```

#### **20) Church Table**

```
CREATE TABLE Church(Org_Name Varchar(20) PRIMARY KEY, FOREIGN KEY(Org_Name) REFERENCES  
External_Org(Org_Name), R_affiliation Varchar(20) NOT NULL);
```

#### **21) External\_Donor Table**

```
CREATE TABLE External_Donor(Org_Name Varchar(20) PRIMARY KEY, FOREIGN KEY(Org_Name)  
REFERENCES External_Org(Org_Name), Anonymous Varchar(20) NOT NULL);
```

#### **22) Ex\_check Table**

```
CREATE TABLE Ex_check(Org_Name Varchar(20), FOREIGN KEY(Org_Name) REFERENCES External_Org  
(Org_Name), Ex_Date DATE, Ex_Amount INT, Ex_Type Varchar(30), Campaign_Name Varchar(30)  
, Cheque_Number INT NOT NULL, PRIMARY KEY(Org_Name, Ex_Date, Ex_Amount, Ex_Type,  
Campaign_Name));
```

#### **23) Ex\_creditcard Table**

```
CREATE TABLE Ex_creditcard(Org_Name Varchar(20), FOREIGN KEY(Org_Name) REFERENCES External_Org  
(Org_Name), ExCC_Date DATE, ExCC_Amount INT, ExCC_Type Varchar(30), ExCC_Campaign_Name  
Varchar(30), ExCard_number INT NOT NULL, ExCard_Type Varchar(20) NOT NULL, Ex_Exp_Date DATE NOT NULL,  
PRIMARY KEY(Org_Name, ExCC_Date, ExCC_Amount, ExCC_Type, ExCC_Campaign_Name));
```

## Screenshots of table creation in SQL

### Person

-----CREATE TABLES-----

```
CREATE Table Person (SSN INT PRIMARY KEY, Name Varchar(30) NOT NULL, Birthdate DATE NOT NULL, Race Varchar(20) NOT NULL, Gender Varchar(10) NOT NULL, Profession Varchar(20) NOT NULL, Mail_address Varchar(100), Email_address Varchar(30), Home_phone INT NOT NULL, Work_phone INT NOT NULL, Cell_phone INT NOT NULL, Mailing_list INT NOT NULL);
```

ages

3:56:06 AM Started executing query at Line 29  
Commands completed successfully.  
Total execution time: 00:00:00.185

### External\_Org

```
CREATE TABLE External_Org (Org_Name Varchar(20) PRIMARY KEY, Org_address Varchar(30) NOT NULL, Org_Number INT NOT NULL, Org_Contact Varchar(20) NOT NULL);
```

ages

3:56:51 AM Started executing query at Line 35  
Commands completed successfully.  
Total execution time: 00:00:00.163

### Client

```
CREATE TABLE Client (SSN INT PRIMARY KEY, Doctor_Name Varchar(30) NOT NULL, Attor_Name Varchar(30) NOT NULL, Doctor_phone INT NOT NULL, Attor_phone INT NOT NULL, Date_assigned DATE NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN));
```

s

3:57:51 AM Started executing query at Line 35  
Commands completed successfully.  
Total execution time: 00:00:00.163

### Employees

```
CREATE TABLE Employees (SSN INT PRIMARY KEY, E_Salary FLOAT NOT NULL, M_Status Varchar(20), HireDate DATE NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN));
```

ges

3:58:27 AM Started executing query at Line 48  
Commands completed successfully.  
Total execution time: 00:00:00.084

## Volunteers

```
CREATE TABLE Volunteers (SSN INT PRIMARY KEY, Joining_Date DATE NOT NULL,  
Training_Date DATE NOT NULL, Training_Loc Varchar(30) NOT NULL,  
FOREIGN KEY (SSN) REFERENCES PERSON(SSN));
```

es

9:42 AM Started executing query at Line 55  
Commands completed successfully.  
Total execution time: 00:00:00.087

## Donors

```
CREATE TABLE Donors (SSN INT PRIMARY KEY, Anonymous Varchar(30) NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN));
```

es

0:07 AM Started executing query at Line 59  
Commands completed successfully.  
Total execution time: 00:00:00.062

## Emergency details

```
CREATE TABLE Emergency_Details (SSN INT, Emer_Name Varchar(30), Contact_Info INT, Relationship Varchar(30),  
PRIMARY KEY (Emer_Name, Contact_Info, Relationship),  
FOREIGN KEY (SSN) REFERENCES PERSON(SSN));
```

jes

00:39 AM Started executing query at Line 62  
Commands completed successfully.  
Total execution time: 00:00:00.089

## Needs

```
CREATE TABLE Needs (SSN INT, Type Varchar(30),  
Imp_Value INT NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN),  
CONSTRAINT Value_check CHECK (Imp_Value between 1 and 10), PRIMARY KEY (SSN, Type, Imp_Value));
```

es

1:37 AM Started executing query at Line 66  
Commands completed successfully.  
Total execution time: 00:00:00.136

## Insurance\_Policy

```
CREATE TABLE Insurance_Policy (SSN INT NOT NULL, Policy_id INT PRIMARY KEY, Provider_id INT NOT NULL, Provider_address Varchar(100) NOT NULL, Type VARCHAR(30) NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN));
```

es

2:18 AM Started executing query at Line 70  
Commands completed successfully.  
Total execution time: 00:00:00.065

## Teams

```
CREATE TABLE Teams (T_Name Varchar(30) PRIMARY KEY, Type Varchar(20) NOT NULL, Date_T INT NOT NULL);
```

es

3:13 AM Started executing query at Line 75  
Commands completed successfully.  
Total execution time: 00:00:00.070

## Cares

```
CREATE TABLE Cares (SSN INT, T_Name Varchar(30), C_ActiveState INT NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN), FOREIGN KEY (T_Name) REFERENCES Teams(T_Name));
```

es

3:41 AM Started executing query at Line 79  
Commands completed successfully.  
Total execution time: 00:00:00.062

## Serves

```
CREATE TABLE Serves (SSN INT, FOREIGN KEY (SSN) REFERENCES PERSON(SSN), T_Name Varchar(30), FOREIGN KEY (T_Name) REFERENCES Teams(T_Name), Hours INT NOT NULL, Active_State INT NOT NULL, Specific_Month Varchar(20) NOT NULL);
```

es

04:22 AM Started executing query at Line 86  
Commands completed successfully.  
Total execution time: 00:00:00.067

## Reports

```
CREATE TABLE Reports (SSN INT, T_Name Varchar(30),  
R_DATE DATE NOT NULL, Desc_1 Varchar(30) NOT NULL, FOREIGN KEY (SSN) REFERENCES PERSON(SSN),  
FOREIGN KEY (T_Name) REFERENCES Teams(T_Name), PRIMARY KEY (T_Name, SSN));
```

iges

10:06:27 AM Started executing query at Line 93  
Commands completed successfully.  
Total execution time: 00:00:00.070

## Expenses

```
CREATE TABLE Expenses (SSN INT, FOREIGN KEY (SSN) REFERENCES PERSON(SSN), Amount FLOAT NOT NULL,  
Desc_1 Varchar(30) NOT NULL, Expense_Date INT NOT NULL, PRIMARY KEY (Amount, Desc_1, Expense_Date, SSN));
```

s

7:33 AM Started executing query at Line 99  
Commands completed successfully.  
Total execution time: 00:00:00.080

## Check

```
CREATE TABLE Check_1 (SSN INT, FOREIGN KEY (SSN) REFERENCES PERSON(SSN),  
C_Date DATE, Amount FLOAT, D_Type Varchar(30), Campaign_Name Varchar(30), Cheque_Number INT NOT NULL,  
PRIMARY KEY (SSN, C_Date, Amount, D_Type, Campaign_Name, Cheque_Number));
```

es

10:25 AM Started executing query at Line 104  
Commands completed successfully.  
Total execution time: 00:00:00.065

## Credit Card

```
CREATE TABLE Credit_Card (SSN INT, FOREIGN KEY (SSN) REFERENCES PERSON(SSN), CC_Date DATE, CC_Amount FLOAT, D_Type Varchar(30),  
Campaign_Name Varchar(30), Card_Number INT NOT NULL,  
Card_Type Varchar(20) NOT NULL, Exp_Date DATE NOT NULL,  
PRIMARY KEY (SSN, CC_Date, CC_Amount, D_Type, Campaign_Name));
```

es

8:46 AM Started executing query at Line 109  
Commands completed successfully.  
Total execution time: 00:00:00.120

## Sponsors

```
CREATE TABLE Sponsors (T_Name Varchar(30), FOREIGN KEY (T_Name) REFERENCES Teams(T_Name),  
Org_Name Varchar(20), FOREIGN KEY (Org_Name) REFERENCES External_Org (Org_Name));
```

jes

09:30 AM Started executing query at Line 112  
Commands completed successfully.  
Total execution time: 00:00:00.066

## Affiliated

```
CREATE TABLE Affiliated (SSN INT, FOREIGN KEY (SSN) REFERENCES PERSON(SSN),  
Org_Name Varchar(20),  
FOREIGN KEY (Org_Name) REFERENCES External_Org (Org_Name), PRIMARY KEY (SSN));
```

jes

10:17 AM Started executing query at Line 117  
Commands completed successfully.  
Total execution time: 00:00:00.062

## Business

```
CREATE TABLE Business (Org_Name Varchar(20) PRIMARY KEY,  
FOREIGN KEY (Org_Name) REFERENCES External_Org (Org_Name),  
Type Varchar(30) NOT NULL, Size INT NOT NULL, Website Varchar(50) NOT NULL);
```

es

10:51 AM Started executing query at Line 123  
Commands completed successfully.  
Total execution time: 00:00:00.065

## Church

```
CREATE TABLE Church (Org_Name Varchar(20) PRIMARY KEY,  
FOREIGN KEY (Org_Name) REFERENCES External_Org (Org_Name),  
R_affiliation Varchar(20) NOT NULL);
```

jes

11:21 AM Started executing query at Line 129  
Commands completed successfully.  
Total execution time: 00:00:00.058



## External donor

```
CREATE TABLE External_Donor (Org_Name Varchar (20) PRIMARY KEY, FOREIGN KEY (Org_Name) REFERENCES External_Org (Org_Name), Anonymous Varchar(20) NOT NULL);
```

### ages

1:11:54 AM Started executing query at Line 135  
Commands completed successfully.  
Total execution time: 00:00:00.066

## Ex\_Check

```
CREATE TABLE Ex_check (Org_Name Varchar (20), FOREIGN KEY (Org_Name) REFERENCES External_Org (Org_Name), Ex_Date DATE, Ex_Amount INT, Ex_Type Varchar (30), Campaign_Name Varchar(30), Cheque_Number INT NOT NULL, PRIMARY KEY (Org_Name, Ex_Date, Ex_Amount, Ex_Type, Campaign_Name));
```

### ages

.:12:22 AM Started executing query at Line 141  
Commands completed successfully.  
Total execution time: 00:00:00.064

## Ex\_Credit

```
CREATE TABLE Ex_creditcard (Org_Name Varchar (20), FOREIGN KEY (Org_Name) REFERENCES External_Org (Org_Name), ExCC_Date DATE, ExCC_Amount INT, ExCC_Type Varchar(30), ExCCCampaign_Name Varchar(30), ExCard_number INT NOT NULL, ExCard_Type Varchar (20) NOT NULL, Ex_Exp_Date DATE NOT NULL, PRIMARY KEY (Org_Name, ExCC_Date, ExCC_Amount, ExCC_Type, ExCCCampaign_Name));
```

### ages

3:20 AM Started executing query at Line 148  
Commands completed successfully.  
Total execution time: 00:00:00.062

## Special Index creation

### Index on Type and Imp\_Value on Table Needs

```
CREATE INDEX Type_V on Needs(Type,Imp_Value);
```

### Index on Type and Type on Table Insurance\_Policy

```
CREATE INDEX I_Type on Insurance_Policy(Type);
```

### Index on Type and Date on Table Teams

```
CREATE INDEX T_Date on Teams(Date_T);
```

### Index on Type and Team Name on Table Cares

```
CREATE INDEX C_TName on Cares(T_Name);
```

### Index on Type and SSN on Table Cares

```
CREATE INDEX C_SSN on Cares(SSN);
```

### Index on Type and Expense Date on Table Expenses

```
CREATE INDEX E_Date on Expenses(Expense_Date);
```

### Index on Type and Org\_Name on Table Sponsors

```
CREATE INDEX Sp_OrgName on Sponsors(Org_Name);
```

### Index on Type and Team Name on Table Serves

```
CREATE INDEX T_Index on Serves(T_Name);
```

## Screenshot:

```
DROP INDEX IF EXISTS Needs.Type_V;
CREATE INDEX Type_V on Needs(Type,Imp_Value);
DROP INDEX IF EXISTS Insurance_Policy.I_Type;
CREATE INDEX I_Type on Insurance_Policy(Type);
DROP INDEX IF EXISTS Teams.T_Date;
CREATE INDEX T_Date on Teams(Date_T);
DROP INDEX IF EXISTS Cares.C_TName;
CREATE INDEX C_TName on Cares(T_Name);
DROP INDEX IF EXISTS Cares.C_SSN;
CREATE INDEX C_SSN on Cares(SSN);
DROP INDEX IF EXISTS Expense.E_Date;
CREATE INDEX E_Date on Expenses(Expense_Date);
DROP INDEX IF EXISTS Sponsors.Sp_OrgName;
CREATE INDEX Sp_OrgName on Sponsors(Org_Name);
DROP INDEX IF EXISTS Serves.T_Index;
CREATE INDEX T_Index on Serves(T_Name);
```

ges

```
:21:59 AM Started executing query at Line 157
Commands completed successfully.
Total execution time: 00:00:00.220
```

## Task 5

### 5.1 SQL statements and Transact SQL stored procedures

```
--To add a Person
DROP PROCEDURE IF EXISTS insertintoPerson
GO
CREATE PROCEDURE insertintoPerson
--Parameters to be taken as input are declared
@SSN INT ,@Name VARCHAR(30) ,
@Birthdate DATE,
@Race VARCHAR(20) ,@Gender VARCHAR(10),
@Profession VARCHAR(20) ,
@Mail_address Varchar(100),@Email_address Varchar(30),@Home_phone INT,
@Work_phone INT,@Cell_phone INT,@Mailing_list INT
AS
BEGIN
    INSERT INTO Person VALUES(@SSN,@Name,@Birthdate,@Race,
    @Gender,@Profession,@Mail_address,@Email_address,@Home_phone,@Work_phone,@Cell_phone,@Mailing_list);
END
GO
```

#### For Option 1)

```
--FOR OPTION-1
DROP PROCEDURE IF EXISTS insertintoTeam
GO
CREATE PROCEDURE insertintoTeam
--Parameters to be taken as input are declared
@T_Name VARCHAR(30) ,
@Type VARCHAR(20) ,
@Date_T INT
AS
BEGIN
    INSERT INTO Teams VALUES(@T_Name,@Type,@Date_T);
END
GO
```

#### For Option 2)

```
--FOR OPTION-2
DROP PROCEDURE IF EXISTS insertintoClient
GO
CREATE PROCEDURE insertintoClient
--Parameters to be taken as input are declared
@SSN INT,
@Doctor_Name VARCHAR(30) ,
@Attor_Name VARCHAR(30) ,
@Doctor_phone INT ,
@Attor_phone INT ,
@Date_assigned DATE
AS
BEGIN
    INSERT INTO Client VALUES(@SSN,@Doctor_Name,@Attor_Name,@Doctor_phone,@Attor_phone,@Date_assigned);
END
```

```

DROP PROCEDURE IF EXISTS insertintoCare
GO
CREATE PROCEDURE insertintoCare
--Parameters to be taken as input are declared
@SSN INT,
@T_Name VARCHAR(30),
@C_ActiveState INT
AS
BEGIN
    INSERT INTO Cares VALUES(@SSN,@T_Name,@C_ActiveState);
END
GO

```

### For Option 3)

```

DROP PROCEDURE IF EXISTS insertintoVolunteer
GO
CREATE PROCEDURE insertintoVolunteer
--Parameters to be taken as input are declared
@SSN INT,
@Joining_Date DATE,
@Training_Date DATE,
@Training_Loc Varchar(30)
AS
✓ BEGIN
    INSERT INTO Volunteers VALUES(@SSN,@Joining_Date,@Training_Date,@Training_Loc);
END
GO

```

### Useful for Option 4 and 3 as well)

```

DROP PROCEDURE IF EXISTS insertintoServes
GO
CREATE PROCEDURE insertintoServes
--Parameters to be taken as input are declared
@SSN INT,
@T_Name VARCHAR(30),
@Hours INT,
@ActiveState INT,
@Specific_Month Varchar(20)
AS
BEGIN
    INSERT INTO Serves VALUES(@SSN,@T_Name,@Hours,@ActiveState,@Specific_Month);
END
GO

```

### For Option 5)

```
DROP PROCEDURE IF EXISTS insertintoEmployee
GO
CREATE PROCEDURE insertintoEmployee
--Parameters to be taken as input are declared
@SSN INT,
@Salary FLOAT,
@M_Status VARCHAR(20),
@Hire_Date DATE
AS
BEGIN
| INSERT INTO Employees VALUES(@SSN,@Salary,@M_Status,@Hire_Date);
END
GO

DROP PROCEDURE IF EXISTS insertintoReports
GO
CREATE PROCEDURE insertintoReports
--Parameters to be taken as input are declared
@SSN INT,
@T_Name VARCHAR(30),
@R_DATE DATE,
@Desc_1 VARCHAR(30)
AS
BEGIN
| INSERT INTO Reports VALUES(@SSN,@T_Name,@R_DATE,@Desc_1);
END
GO
```

### For Option 6)

```
DROP PROCEDURE IF EXISTS insertintoExpense
GO
CREATE PROCEDURE insertintoExpense
--Parameters to be taken as input are declared
@SSN INT,
@Amount FLOAT,
@Desc_1 VARCHAR(30),
@Expense_Date INT
AS
BEGIN
| INSERT INTO Expenses VALUES(@SSN,@Amount,@Desc_1,@Expense_Date);
END
GO
```

### For Option 7)

```
DROP PROCEDURE IF EXISTS insertintoExternal_Org
GO
CREATE PROCEDURE insertintoExternal_Org
--Parameters to be taken as input are declared
@Org_Name Varchar (20),
@Org_address Varchar (30),
@Org_Number INT,
@Org_Contact Varchar(20)

AS
BEGIN
    INSERT INTO External_Org VALUES(@Org_Name,@Org_address,@Org_Number,@Org_Contact);
END
GO

DROP PROCEDURE IF EXISTS insertintoSponsors
GO
CREATE PROCEDURE insertintoSponsors
--Parameters to be taken as input are declared
@T_Name VARCHAR(30),
@Org_Name VARCHAR(20)

AS
BEGIN
    INSERT INTO Sponsors VALUES(@T_Name,@Org_Name);
END
GO
```

### For Option 8)

```
DROP PROCEDURE IF EXISTS insertintoDonor
GO
CREATE PROCEDURE insertintoDonor
--Parameters to be taken as input are declared
@SSN INT,
@Anonymous VARCHAR(30)

AS
BEGIN
    INSERT INTO Donors VALUES(@SSN,@Anonymous);
END
GO
```

```

DROP PROCEDURE IF EXISTS insertintoCheck
GO
CREATE PROCEDURE insertintoCheck
--Parameters to be taken as input are declared
@SSN INT,
@Date DATE,
@DAmount FLOAT,
@DTYPE VARCHAR(30),
@CampaignName VARCHAR(30),
@ChequeNumber INT
AS
BEGIN
    INSERT INTO Check_1 VALUES(@SSN,@Date,@DAmount,@DTYPE,@CampaignName,@ChequeNumber);
END
GO

```

```

DROP PROCEDURE IF EXISTS insertintoCredit
GO
CREATE PROCEDURE insertintoCredit
--Parameters to be taken as input are declared
@SSN INT,
@Date DATE,
@DAmount FLOAT,
@DTYPE VARCHAR(30),
@CampaignName VARCHAR(30),
@CardNumber INT,
@CardType VARCHAR(20),
@ExpDate DATE
AS
BEGIN
    INSERT INTO Credit_Card VALUES(@SSN,@Date,@DAmount,@DTYPE,@CampaignName,@CardNumber,@CardType,@ExpDate);
END
GO

```

### For Option 9)

```

DROP PROCEDURE IF EXISTS insertintoExternalDonor
GO
CREATE PROCEDURE insertintoExternalDonor
--Parameters to be taken as input are declared
@Org_Name VARCHAR(20),
@Anonymous VARCHAR(20)
AS
BEGIN
    INSERT INTO External_Donor VALUES(@Org_Name,@Anonymous);
END
GO

```

```

DROP PROCEDURE IF EXISTS insertintoExCheck
GO
CREATE PROCEDURE insertintoExCheck
--Parameters to be taken as input are declared
@Org_Name Varchar (20),
@Date DATE,
@DAmount FLOAT,
@DTYPE VARCHAR(30),
@CampaignName VARCHAR(30),
@ChequeNumber INT
AS
BEGIN
|   INSERT INTO Ex_check VALUES(@Org_Name,@Date,@DAmount,@DTYPE,@CampaignName,@ChequeNumber);
END
GO

```

```

DROP PROCEDURE IF EXISTS insertintoExCredit
GO
CREATE PROCEDURE insertintoExCredit
--Parameters to be taken as input are declared
@Org_Name Varchar (20),
@Date DATE,
@DAmount FLOAT,
@DTYPE VARCHAR(30),
@CampaignName VARCHAR(30),
@CardNumber INT,
@CardType VARCHAR(20),
@ExpDate DATE
AS
BEGIN
|   INSERT INTO Ex_creditcard VALUES(@Org_Name,@Date,@DAmount,@DTYPE,@CampaignName,@CardNumber,@CardType,@ExpDate);
END
GO

```



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

```
1 //Import statements
2 import java.io.BufferedReader;
3 import java.io.BufferedWriter;
4 import java.io.DataInputStream;
5 import java.io.FileInputStream;
6 import java.io.FileNotFoundException;
7 import java.io.FileWriter;
8 import java.io.IOException;
9 import java.io.InputStreamReader;
10 import java.sql.CallableStatement;
11 import java.sql.Connection;
12 import java.sql.Date;
13 import java.sql.Statement;
14 import java.sql.Time;
15 import java.util.InputMismatchException;
16 import java.util.Scanner;
17 import java.sql.ResultSet;
18 import java.sql.SQLException;
19 import java.sql.DriverManager;
20 import java.sql.PreparedStatement;
21 //Class
22 public class finalproject {
23     // Database credentials
24     final static String HOSTNAME = "chal0018-sql-server.database.windows.net";
25     final static String DBNAME = "HM-2";
26     final static String USERNAME = "chal0018";
27     final static String PASSWORD = "*****";
28     // Database connection string
29     final static String URL = String.format(
30         "jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;trustServerCertificate=false;hostNameInCertificate=%s";
31         HOSTNAME, DBNAME, USERNAME, PASSWORD);
32     // Query templates
33     final static String QUERY_TEMPLATE_1 = "{call insertintoTeam(?,?,?)}";
34     final static String QUERY_TEMPLATE_2 = "{call insertintoClient(?,?,?,?)}";
35     final static String QUERY_TEMPLATE_3 = "{call insertintoCare(?,?,?)}";
36     final static String QUERY_TEMPLATE_4 = "{call insertintoPerson(?,?,?,?,?,?)}";
37     final static String QUERY_TEMPLATE_5 = "{call insertintoVolunteer(?,?,?)}";
38     final static String QUERY_TEMPLATE_6 = "{call insertintoServes(?,?,?,?)}";
39     final static String QUERY_TEMPLATE_7 = "{call insertintoEmployee(?,?,?,?)}";
40     final static String QUERY_TEMPLATE_8 = "{call insertintoReports(?,?,?,?)}";
41     final static String QUERY_TEMPLATE_9 = "{call insertintoExpense(?,?,?,?)}";
42     final static String QUERY_TEMPLATE_10 = "{call insertintoExternal_Org(?,?,?,?)}";
43     final static String QUERY_TEMPLATE_11 = "{call insertintoSponsors(?,?,?)}";
44     final static String QUERY_TEMPLATE_12 = "{call insertintoDonor(?,?,?)}";
45     final static String QUERY_TEMPLATE_13 = "{call insertintoCheck(?,?,?,?,?,?)}";
46     final static String QUERY_TEMPLATE_14 = "{call insertintoCredit(?,?,?,?,?,?)}";
47     final static String QUERY_TEMPLATE_15 = "{call insertintoExternalDonor(?,?,?)}";
48     final static String QUERY_TEMPLATE_16 = "{call insertintoExCheck(?,?,?,?,?,?)}";
49     final static String QUERY_TEMPLATE_17 = "{call insertintoExCredit(?,?,?,?,?,?)}";
50     final static String QUERY_TEMPLATE_18 = "select * from Employees";
51     final static String QUERY_TEMPLATE_19 = "select * from Client";
52     // User input prompt
53     final static String PROMPT = "\nPlease select one of the options below: \n" + "1) Insert new Team: \n"
54         + "2) Insert new Client and associate him/her to Team \n" + "3) Insert new Volunteer and associate him/her to Team \n"
55         + "4) Enter number of hrs volunteer worked for a Team \n" + "5) Insert new Employee and associate him/her to Team \n" +
56         "6) Enter a new Expense charged by Employee \n" +
57         "7) Insert new Organization and associate it to one/more Teams \n" +
58         "8) Insert new Donors and associate it to one/more donations \n" +
59         "9) Insert new Organization and associate it to one/more donations \n" +
60         "10) Retrieve Doctor Name and Phone for particular client \n" +
61         "11) Retrieve total amount of expenses for particular period of an employee date format-YYYYMMDD \n" +
62         "12) Retrieve list of Volunteers that are members of team that support particular client \n" +
63         "13) Retrieve Name,Contact info of Client supported by an Org with name starts between B & K \n" +
64         "14) yet to implement \n" +
65         "15) Retrieve Team Name that are found after a particular date(yyyymmdd) \n" +
66         "16) Update Salary of Employee who has more than one team reporting \n" +
67         "17) Delete Employee who does not have Insurance and type is transportation \n" +
68         "18) Import file \n" + "19) Export File \n" + "20) Exit!";
69
70     public static void main(String[] args) throws SQLException{
71         System.out.println("Welcome to the sample application!");
72         final Scanner sc = new Scanner(System.in); // Scanner is used to collect the user input
73         String option = ""; // Initialize user option selection as nothing
74         Connection conn = DriverManager.getConnection(URL); //establish connection
75         Statement stmt = conn.createStatement(); //to execute sql queries
76         BufferedReader br = new BufferedReader(new InputStreamReader(System.in)); //used for export/ import read input from user
77     }
```

```

78 //Insert a person at the start
79 System.out.println("Please enter SSN:");
80 final int SSN_P= sc.nextInt();
81 sc.nextLine();
82 System.out.println("Please enter Person Name:");
83 final String P_Name = sc.nextLine();
84 System.out.println("Please enter Birth Date:");
85 final String Birthdate=sc.nextLine();
86 Date Bdate=Date.valueOf(Birthdate);
87 System.out.println("Please enter Race:");
88 final String Race = sc.nextLine();
89 System.out.println("Please enter Gender:");
90 final String Gender = sc.nextLine();
91 System.out.println("Please enter Profession:");
92 final String Profession = sc.nextLine();
93 System.out.println("Please enter Mail address:");
94 final String Mail_address = sc.nextLine();
95 System.out.println("Please enter Email address:");
96 final String Email_address = sc.nextLine();
97 System.out.println("Please enter Home_phone:");
98 final int Home_phone= sc.nextInt();
99 System.out.println("Please enter Work_phone:");
100 final int Work_phone= sc.nextInt();
101 System.out.println("Please enter Cell_phone:");
102 final int Cell_phone= sc.nextInt();
103 System.out.println("Please enter Mailing_list state:");
104 final int Mailing_list= sc.nextInt();
105
106 try (final Connection connection = DriverManager.getConnection(URL)) {
107     try (final CallableStatement statement = connection.prepareCall(QUERY_TEMPLATE_4)) {
108         // Populate the query template with the data collected from the user
109         statement.setInt(1,SSN_P );
110         statement.setString(2, P_Name);
111         statement.setDate(3, Bdate);
112         statement.setString(4, Race);
113         statement.setString(5, Gender);
114         statement.setString(6, Profession);
115         statement.setString(7, Mail_address);
116         statement.setString(8, Email_address);
117
118         statement.setInt(9, Home_phone);
119         statement.setInt(10, Work_phone);
120         statement.setInt(11, Cell_phone);
121         statement.setInt(12, Mailing_list);
122         System.out.println("Dispatching the query...");
123         // Actually execute the populated query
124         final int rows_inserted = statement.executeUpdate();
125         System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
126     }
127 }
128
129 while (!option.equals("20")) { // Ask user for options until option 20 is selected
130     System.out.println(PROMPT); // Print the available options
131     option = sc.next(); // Read in the user option selection
132     switch (option) { // Switch between different options
133
134     case "1": // Insert a new Team option
135         System.out.println("Please enter Team Name:");
136         try {
137             sc.nextLine();
138             final String T_Name = sc.nextLine(); // Read in the user input of Team Name
139             System.out.println("Please enter Team Type:");
140             // We call nextLine to consume that newline character, so that subsequent
141             // nextLine doesn't return nothing.
142             final String Type = sc.nextLine(); // Read in user input of faculty Name (white-spaces allowed).
143             System.out.println("Please Date assigned:");
144             final int date_a=sc.nextInt();
145             System.out.println("Connecting to the database...");
146             // Get a database connection and prepare a query statement
147             try (final Connection connection = DriverManager.getConnection(URL)) {
148                 try (final CallableStatement statement = connection.prepareCall(QUERY_TEMPLATE_1)) {
149                     // Populate the query template with the data collected from the user
150                     statement.setString(1,T_Name );
151                     statement.setString(2, Type);
152                     statement.setInt(3, date_a);
153                     System.out.println("Dispatching the query...");
154                     // Actually execute the populated query
155                     final int rows_inserted = statement.executeUpdate();

```

```

156     }
157 }
158 }
159 //Error handling
160 catch(InputMismatchException e)
161 {
162     System.out.println("Caught Input mismatch -- ");
163     sc.nextLine();
164 }
165 }
166 catch (Exception e)
167 {
168     System.out.println("Couldn't execute the above query!");
169     System.out.println("Reason -- ");
170     System.out.println(e.getMessage());
171     System.out.println("Try again !!!");
172 }
173 break;
174 case "2": // Insert a new Client
175     try {
176         System.out.println("Please enter SSN:");
177         final int SSN = sc.nextInt();
178         sc.nextLine();
179         System.out.println("Please enter Doctor name:");
180         final String Doctor_Name = sc.nextLine();
181         System.out.println("Please enter Attor Name:");
182         final String Attor_Name= sc.nextLine();
183         System.out.println("Please Doctor Phone:");
184         final int Doctor_phone = sc.nextInt();
185         System.out.println("Please enter Attor Phone:");
186         final int Attor_Phone= sc.nextInt();
187         sc.nextLine();
188         System.out.println("Please Date assigned:");
189         final String date_a=sc.nextLine();
190         Date date=Date.valueOf(date_a);//converting string into sql date
191         System.out.println("Connecting to the database...");
192         // Get a database connection and prepare a query statement
193         try (final Connection connection = DriverManager.getConnection(URL)) {

194             try (final CallableStatement statement = connection.prepareCall(QUERY_TEMPLATE_2))
195             {
196                 // Populate the query template with the data collected from the user
197                 statement.setInt(1,SSN );
198                 statement.setString(2, Doctor_Name);
199                 statement.setString(3, Attor_Name);
200                 statement.setInt(4, Doctor_phone);
201                 statement.setInt(5, Attor_Phone);
202                 statement.setDate(6, date);
203                 System.out.println("Dispatching the query...");
204                 // Actually execute the populated query
205                 final int rows_inserted = statement.executeUpdate();
206                 System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
207             }
208         }
209     }
210 }
211 catch(InputMismatchException e)
212 {
213     System.out.println("Caught Input mismatch");
214     sc.nextLine();
215     break;
216 }
217 catch (Exception e)
218 {
219     System.out.println("Couldn't execute the above query!");
220     System.out.println("Reason -- ");
221     System.out.println(e.getMessage());
222     System.out.println("Try again !!!");
223 }
224
225
226 //for care
227 System.out.println("Number of teams to associate to");
228 int count=sc.nextInt();
229 sc.nextLine();
230 while (count!=0) {
231     System.out.println("Please enter SSN:");
232     final int SSN = sc.nextInt();

```

```

233         sc.nextLine();
234         System.out.println("Please enter Team Name to which you want to associate:");
235         final String Team_Name= sc.nextLine();
236         int C_ActiveState=0;
237         try (final Connection connection = DriverManager.getConnection(URL)) {
238             try (final CallableStatement statement1 = connection.prepareCall(QUERY_TEMPLATE_3))
239             {
240                 // Populate the query template with the data collected from the user
241                 statement1.setInt(1,SSN );
242                 statement1.setString(2, Team_Name);
243                 statement1.setInt(3, C_ActiveState);
244                 System.out.println("Dispatching the query...");
245                 // Actually execute the populated query
246                 final int rows_inserted = statement1.executeUpdate();
247                 System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
248             }
249         }
250         count--;
251         break;
252     }
253
254     case "3": // Insert a new Volunteer
255
256         try {
257             System.out.println("Please enter SSN:");
258             final int SSN = sc.nextInt();
259             sc.nextLine();
260             System.out.println("Please enter Joining Date:");
261             final String date_J=sc.nextLine();
262             Date date=Date.valueOf(date_J);
263             System.out.println("Please Training Date:");
264             final String date_T = sc.nextLine();
265             Date date1=Date.valueOf(date_T);
266             System.out.println("Please enter Training Loc :");
267             final String Training_Loc = sc.nextLine();
268
269             //converting string into sql date
270             System.out.println("Connecting to the database...");
271
272             // Get a database connection and prepare a query statement
273             try (final Connection connection = DriverManager.getConnection(URL)) {
274                 try (final CallableStatement statement = connection.prepareCall(QUERY_TEMPLATE_5))
275                 {
276                     // Populate the query template with the data collected from the user
277                     statement.setInt(1,SSN );
278                     statement.setDate(2, date);
279                     statement.setDate(3, date1);
280                     statement.setString(4, Training_Loc);
281                     System.out.println("Dispatching the query...");
282                     // Actually execute the populated query
283                     final int rows_inserted = statement.executeUpdate();
284                     System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
285                 }
286             }
287         }
288         catch(InputMismatchException e)
289         {
290             System.out.println("Caught Input mismatch");
291             sc.nextLine();
292             break;
293         }
294         catch (Exception e)
295         {
296             System.out.println("Couldn't execute the above query!");
297             System.out.println("Reason -- ");
298             System.out.println(e.getMessage());
299             System.out.println("Try again !!!");
300         }
301
302     //for SERVES
303     System.out.println("Number of teams to associate to");
304     int count1=sc.nextInt();
305     sc.nextLine();
306     while (count1!=0) {
307         System.out.println("Please enter SSN:");
308         final int SSN = sc.nextInt();
309         sc.nextLine();
310

```

```

311         System.out.println("Please enter Team Name to which you want to associate:");
312         final String Team_Name= sc.nextLine();
313         System.out.println("Please enter HOURS:");
314         final int hours = sc.nextInt();
315         sc.nextLine();
316         int C_ActiveState=0;
317         System.out.println("Please enter Specific month:");
318         final String month=sc.nextLine();
319         try (final Connection connection = DriverManager.getConnection(URL)) {
320             try (final CallableStatement statement1 = connection.prepareCall(QUERY_TEMPLATE_6))
321             {
322                 // Populate the query template with the data collected from the user
323                 statement1.setInt(1,SSN );
324                 statement1.setString(2, Team_Name);
325                 statement1.setInt(3, hours);
326                 statement1.setInt(4, C_ActiveState);
327                 statement1.setString(5, month);
328                 System.out.println("Dispatching the query...");
329                 // Actually execute the populated query
330                 final int rows_inserted = statement1.executeUpdate();
331                 System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
332             }
333         }
334         count1--;
335         break;
336
337     case "4":
338     try {
339         //for SERVES
340
341         System.out.println("Please enter SSN:");
342         final int SSN = sc.nextInt();
343         sc.nextLine();
344         System.out.println("Please enter Team Name to which you want to associate:");
345         final String Team_Name= sc.nextLine();
346         System.out.println("Please enter HOURS:");
347         final int hours = sc.nextInt();
348         sc.nextLine();

```

```

349         int C_ActiveState=0;
350         System.out.println("Please enter Specific month:");
351         final String month=sc.nextLine();
352         try (final Connection connection = DriverManager.getConnection(URL)) {
353             try (final CallableStatement statement1 = connection.prepareCall(QUERY_TEMPLATE_6))
354             {
355                 // Populate the query template with the data collected from the user
356                 statement1.setInt(1,SSN );
357                 statement1.setString(2, Team_Name);
358                 statement1.setInt(3, hours);
359                 statement1.setInt(4, C_ActiveState);
360                 statement1.setString(5, month);
361                 System.out.println("Dispatching the query...");
362                 // Actually execute the populated query
363                 final int rows_inserted = statement1.executeUpdate();
364                 System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
365             }
366         }
367     }
368
369     catch(InputMismatchException e)
370     {
371         System.out.println("Caught Input mismatch");
372         sc.nextLine();
373         break;
374     }
375     catch (Exception e)
376     {
377         System.out.println("Couldn't execute the above query!");
378         System.out.println("Reason -- ");
379         System.out.println(e.getMessage());
380         System.out.println("Try again !!!");
381     }
382     break;
383
384     case "5": // Insert a new Employee
385         // Collect the new Employee data from the user
386
387         try {

```

```

388      System.out.println("Please enter SSN:");
389      final int SSN = sc.nextInt();
390      sc.nextLine();
391      System.out.println("Please enter Salary of Employee:");
392      final Float Salary=sc.nextFloat();
393      sc.nextLine();
394      System.out.println("Please enter Marital Status:");
395      final String Status = sc.nextLine();
396      System.out.println("Please enter Hire Date:");
397      final String date_J=sc.nextLine();
398      Date date=Date.valueOf(date_J);
399
400
401      //converting string into sql date
402      System.out.println("Connecting to the database...");
403      // Get a database connection and prepare a query statement
404      try (final Connection connection = DriverManager.getConnection(URL)) {
405          try (final CallableStatement statement = connection.prepareCall(QUERY_TEMPLATE_7))
406          {
407              // Populate the query template with the data collected from the user
408              statement.setInt(1,SSN );
409              statement.setFloat(2, Salary);
410              statement.setString(3, Status);
411              statement.setDate(4, date);
412              System.out.println("Dispatching the query...");
413              // Actually execute the populated query
414              final int rows_inserted = statement.executeUpdate();
415              System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
416          }
417      }
418  }
419  }
420  catch(InputMismatchException e)
421  {
422      System.out.println("Caught Input mismatch");
423      sc.nextLine();
424      break;
425  }
426
427
428  catch (Exception e)
429  {
430      System.out.println("Couldn't execute the above query!");
431      System.out.println("Reason -- ");
432      System.out.println(e.getMessage());
433      System.out.println("Try again !!!");
434  }
435
436  //for Reports
437  System.out.println("Number of teams to associate to");
438  int count2=sc.nextInt();
439  sc.nextLine();
440  while (count2!=0) {
441      System.out.println("Please enter SSN:");
442      final int SSN = sc.nextInt();
443      sc.nextLine();
444      System.out.println("Please enter Team Name to which you want to associate:");
445      final String Team_Name= sc.nextLine();
446      System.out.println("Please enter Report Date:");
447      final String date_J=sc.nextLine();
448      Date date=Date.valueOf(date_J);
449      System.out.println("Please enter Report Description:");
450      final String Desc=sc.nextLine();
451      try (final Connection connection = DriverManager.getConnection(URL)) {
452          try (final CallableStatement statement1 = connection.prepareCall(QUERY_TEMPLATE_8))
453          {
454              // Populate the query template with the data collected from the user
455              statement1.setInt(1,SSN );
456              statement1.setString(2, Team_Name);
457              statement1.setDate(3, date);
458              statement1.setString(4, Desc);
459              System.out.println("Dispatching the query...");
460              // Actually execute the populated query
461              final int rows_inserted = statement1.executeUpdate();
462              System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
463          }
464      }
465  }

```

```

464         count2--;
465         break;
466
467
468     case "6":
469         try {
470             //for Expense
471             System.out.println("Please enter SSN:");
472             final int SSN = sc.nextInt();
473             sc.nextLine();
474             System.out.println("Please enter Expense Amount:");
475             final Float Expense_A= sc.nextFloat();
476             sc.nextLine();
477             System.out.println("Please enter Expense Description:");
478             final String Description=sc.nextLine();
479             System.out.println("Please enter Expense Date:");
480             final int date_J=sc.nextInt();
481             //Date date=Date.valueOf(date_J);
482             try (final Connection connection = DriverManager.getConnection(URL)) {
483                 try (final CallableStatement statement1 = connection.prepareCall(QUERY_TEMPLATE_9))
484                 {
485                     // Populate the query template with the data collected from the user
486                     statement1.setInt(1,SSN );
487                     statement1.setFloat(2, Expense_A);
488                     statement1.setString(3, Description);
489                     statement1.setInt(4, date_J);
490
491                     System.out.println("Dispatching the query...");
492                     // Actually execute the populated query
493                     final int rows_inserted = statement1.executeUpdate();
494                     System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
495                 }
496             }
497         }
498
499         catch(InputMismatchException e)
500         {
501             System.out.println("Caught Input mismatch");
502             sc.nextLine();
503
504             break;
505         }
506         catch (Exception e)
507         {
508             System.out.println("Couldn't execute the above query!");
509             System.out.println("Reason -- ");
510             System.out.println(e.getMessage());
511             System.out.println("Try again !!!");
512             break;
513
514
515     case "7": // Insert a new Organization
516         // Collect the new Organization data from the user
517         System.out.println("Please enter Organization Name:");
518         try {
519             sc.nextLine();
520             final String Org_Name = sc.nextLine();
521             System.out.println("Please enter Organization Address:");
522             final String Org_Address = sc.nextLine();
523             System.out.println("Please Org Number:");
524             final int Org_Number = sc.nextInt();
525             sc.nextLine();
526             System.out.println("Please enter Org Contact :");
527             final String Org_Contact = sc.nextLine();
528
529             //converting string into sql date
530             System.out.println("Connecting to the database...");
531             // Get a database connection and prepare a query statement
532             try (final Connection connection = DriverManager.getConnection(URL)) {
533                 try (final CallableStatement statement = connection.prepareCall(QUERY_TEMPLATE_10))
534                 {
535                     // Populate the query template with the data collected from the user
536                     statement.setString(1,Org_Name );
537                     statement.setString(2, Org_Address);
538                     statement.setInt(3, Org_Number);
539                     statement.setString(4, Org_Contact);
540                     System.out.println("Dispatching the query...");
541                     // Actually execute the populated query

```



```

541         // Actually execute the populated query
542         final int rows_inserted = statement.executeUpdate();
543         System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
544     }
545 }
546 }
547 }
548 catch(InputMismatchException e)
549 {
550     System.out.println("Caught Input mismatch");
551     sc.nextLine();
552     break;
553 }
554 catch (Exception e)
555 {
556     System.out.println("Couldn't execute the above query!");
557     System.out.println("Reason -- ");
558     System.out.println(e.getMessage());
559     System.out.println("Try again !!!");
560 }
561 }
562
563 //for Sponsor
564 System.out.println("Number of teams to associate to");
565 int count3=sc.nextInt();
566 sc.nextLine();
567 while (count3!=0) {
568
569     System.out.println("Please enter Team Name to which you want to associate:");
570     final String Team_Name= sc.nextLine();
571     System.out.println("Please enter Organization Name:");
572     final String Org_Name = sc.nextLine();
573     try (final Connection connection = DriverManager.getConnection(URL)) {
574         try (final CallableStatement statement1 = connection.prepareCall(QUERY_TEMPLATE_11))
575         {
576             // Populate the query template with the data collected from the user
577             statement1.setString(1,Team_Name );
578             statement1.setString(2, Org_Name);
579             System.out.println("Dispatching the query...");

```

```

580         // Actually execute the populated query
581         final int rows_inserted = statement1.executeUpdate();
582         System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
583     }
584 }
585     count3--;
586     break;
587 }
588 case "8": // Insert a new Donor
589     // Collect the new Donor data from the user
590
591     try {
592         System.out.println("Please enter SSN:");
593         final int SSN = sc.nextInt();
594         sc.nextLine();
595         System.out.println("Please enter Anonymous state of Donor :");
596         final String Anonymous_S = sc.nextLine();
597         //converting string into sql date
598         System.out.println("Connecting to the database...");
599         // Get a database connection and prepare a query statement
600         try (final Connection connection = DriverManager.getConnection(URL)) {
601             try (final CallableStatement statement = connection.prepareCall(QUERY_TEMPLATE_12))
602             {
603                 // Populate the query template with the data collected from the user
604                 statement.setInt(1,SSN );
605                 statement.setString(2,Anonymous_S );
606                 System.out.println("Dispatching the query...");
607                 // Actually execute the populated query
608                 final int rows_inserted = statement.executeUpdate();
609                 System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
610             }
611         }
612     }
613 }
614 catch(InputMismatchException e)
615 {
616     System.out.println("Caught Input mismatch");
617     sc.nextLine();
618     break;

```



```

620         catch (Exception e)
621         {
622             System.out.println("Couldn't execute the above query!");
623             System.out.println("Reason -- ");
624             System.out.println(e.getMessage());
625             System.out.println("Try again !!!");
626         }
627
628
629         //for Donations
630         System.out.println("Number of donations to associate to");
631         int count4=sc.nextInt();
632         sc.nextLine();
633         while (count4!=0) {
634
635             System.out.println("Please enter Donation payment 1:check and 2:Credit:");
636             int Don_type= sc.nextInt();
637             if (Don_type==1) {
638                 System.out.println("Please enter SSN:");
639                 final int SSN = sc.nextInt();
640                 sc.nextLine();
641                 System.out.println("Please enter Date of Donation:");
642                 final String date_J=sc.nextLine();
643                 Date date=Date.valueOf(date_J);
644                 System.out.println("Please enter Donation Amount:");
645                 final float D_amount=sc.nextFloat();
646                 sc.nextLine();
647                 System.out.println("Please enter Donation type:");
648                 final String D_Type = sc.nextLine();
649                 System.out.println("Please enter Campaign Name:");
650                 final String Camp_Name = sc.nextLine();
651                 System.out.println("Please enter Cheque Number:");
652                 final int Check_Num = sc.nextInt();
653                 try (final Connection connection = DriverManager.getConnection(URL)) {
654                     try (final CallableStatement statement1 = connection.prepareCall(QUERY_TEMPLATE_13))
655                     {
656                         // Populate the query template with the data collected from the user
657                         statement1.setInt(1,SSN );
658
659
660                         statement1.setDate(2, date);
661                         statement1.setFloat(3, D_amount);
662                         statement1.setString(4, D_Type);
663                         statement1.setString(5, Camp_Name);
664                         statement1.setInt(6, Check_Num);
665
666                         System.out.println("Dispatching the query...");
667                         // Actually execute the populated query
668                         final int rows_inserted = statement1.executeUpdate();
669                         System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
670                     }
671                 }
672             }
673             else {
674                 System.out.println("Please enter SSN:");
675                 final int SSN = sc.nextInt();
676                 sc.nextLine();
677                 System.out.println("Please enter Date of Donation:");
678                 final String date_J=sc.nextLine();
679                 Date date=Date.valueOf(date_J);
680                 System.out.println("Please enter Donation Amount:");
681                 final float D_amount=sc.nextFloat();
682                 sc.nextLine();
683                 System.out.println("Please enter Donation type:");
684                 final String D_Type = sc.nextLine();
685                 System.out.println("Please enter Campaign Name:");
686                 final String Camp_Name = sc.nextLine();
687                 System.out.println("Please enter Card Number:");
688                 final int Check_Num = sc.nextInt();
689                 sc.nextLine();
690                 System.out.println("Please enter Card Type:");
691                 final String CardType= sc.nextLine();
692                 System.out.println("Please enter Expiry Date:");
693                 final String date_E=sc.nextLine();
694                 Date dateE=Date.valueOf(date_E);
695                 try (final Connection connection = DriverManager.getConnection(URL)) {
696                     try (final CallableStatement statement1 = connection.prepareCall(QUERY_TEMPLATE_14))
697                     {
698                         // Populate the query template with the data collected from the user

```

```

694         try (final CallableStatement statement1 = connection.prepareCall(QUERY_TEMPLATE_14))
695         {
696             // Populate the query template with the data collected from the user
697             statement1.setInt(1, SSN );
698             statement1.setDate(2, date);
699             statement1.setFloat(3, D_amount);
700             statement1.setString(4, D_Type);
701             statement1.setString(5, Camp_Name);
702             statement1.setInt(6, Check_Num);
703             statement1.setString(7, CardType);
704             statement1.setDate(8, dateE);
705
706
707             System.out.println("Dispatching the query...");
708             // Actually execute the populated query
709             final int rows_inserted = statement1.executeUpdate();
710             System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
711         }
712     }
713
714
715     }
716     count4--;
717     break;
718
719
720     case "9": // Insert a new Organization+Donations
721         // Collect the new Organization data from the user
722
723         System.out.println("Please enter Organization Name:");
724         try {
725             sc.nextLine();
726             final String Org_Name = sc.nextLine();
727             System.out.println("Please enter Organization Address:");
728             final String Org_Address = sc.nextLine();
729             System.out.println("Please Org Number:");
730             final int Org_Number = sc.nextInt();
731             sc.nextLine();
732             System.out.println("Please enter Org Contact :");

```

```

733         final String Org_Contact = sc.nextLine();
734
735         //converting string into sql date
736         System.out.println("Connecting to the database...");
737         // Get a database connection and prepare a query statement
738         try (final Connection connection = DriverManager.getConnection(URL)) {
739             try (final CallableStatement statement = connection.prepareCall(QUERY_TEMPLATE_10))
740             {
741                 // Populate the query template with the data collected from the user
742                 statement.setString(1, Org_Name );
743                 statement.setString(2, Org_Address);
744                 statement.setInt(3, Org_Number);
745                 statement.setString(4, Org_Contact);
746                 System.out.println("Dispatching the query...");
747                 // Actually execute the populated query
748                 final int rows_inserted = statement.executeUpdate();
749                 System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
750             }
751         }
752     }
753 }
754 catch (InputMismatchException e)
755 {
756     System.out.println("Caught Input mismatch");
757     sc.nextLine();
758     break;
759 }
760 catch (Exception e)
761 {
762     System.out.println("Couldn't execute the above query!");
763     System.out.println("Reason -- ");
764     System.out.println(e.getMessage());
765     System.out.println("Try again !!!");
766 }
767
768 try {
769     System.out.println("Please enter Organization Name:");
770     final String Organization_N = sc.nextLine();
771     System.out.println("Please enter Anonymous state of Donor :");

```

```

772         final String Anonymous_S = sc.nextLine();
773         //converting string into sql date
774         System.out.println("Connecting to the database...");
775         // Get a database connection and prepare a query statement
776         try (final Connection connection = DriverManager.getConnection(URL)) {
777             try (final CallableStatement statement = connection.prepareCall(QUERY_TEMPLATE_15))
778             {
779                 // Populate the query template with the data collected from the user
780                 statement.setString(1, Organization_N );
781                 statement.setString(2, Anonymous_S );
782                 System.out.println("Dispatching the query...");
783                 // Actually execute the populated query
784                 final int rows_inserted = statement.executeUpdate();
785                 System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
786             }
787         }
788     }
789 }
790 catch (InputMismatchException e)
791 {
792     System.out.println("Caught Input mismatch");
793     sc.nextLine();
794     break;
795 }
796 catch (Exception e)
797 {
798     System.out.println("Couldn't execute the above query!");
799     System.out.println("Reason -- ");
800     System.out.println(e.getMessage());
801     System.out.println("Try again !!!");
802 }
803
804 //for Donations
805 System.out.println("Number of donations to associate to");
806 int count5=sc.nextInt();
807 sc.nextLine();
808 while (count5!=0) {
809
810     System.out.println("Please enter Donation payment 1:check and 2:Credit:");

```

```

811     int Don_type= sc.nextInt();
812     sc.nextLine();
813     if (Don_type==1) {
814         System.out.println("Please enter Org Name:");
815         final String Org_Name = sc.nextLine();
816         System.out.println("Please enter Date of Donation:");
817         final String date_3=sc.nextLine();
818         Date date=Date.valueOf(date_3);
819         System.out.println("Please enter Donation Amount:");
820         final float D_amount=sc.nextFloat();
821         sc.nextLine();
822         System.out.println("Please enter Donation type:");
823         final String D_Type = sc.nextLine();
824         System.out.println("Please enter Campaign Name:");
825         final String Camp_Name = sc.nextLine();
826         System.out.println("Please enter Cheque Number:");
827         final int Check_Num = sc.nextInt();
828         try (final Connection connection = DriverManager.getConnection(URL)) {
829             try (final CallableStatement statement1 = connection.prepareCall(QUERY_TEMPLATE_16))
830             {
831                 // Populate the query template with the data collected from the user
832                 statement1.setString(1, Org_Name );
833                 statement1.setDate(2, date);
834                 statement1.setFloat(3, D_amount);
835                 statement1.setString(4, D_Type);
836                 statement1.setString(5, Camp_Name);
837                 statement1.setInt(6, Check_Num);
838
839                 System.out.println("Dispatching the query...");
840                 // Actually execute the populated query
841                 final int rows_inserted = statement1.executeUpdate();
842                 System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
843             }
844         }
845     }
846     else {
847         System.out.println("Please enter Org Name:");
848         final String Org_Name = sc.nextLine();
849         System.out.println("Please enter Date of Donation:");

```

```

850         final String date_J=sc.nextLine();
851         Date date=Date.valueOf(date_J);
852         System.out.println("Please enter Donation Amount:");
853         final float D_amount=sc.nextFloat();
854         sc.nextLine();
855         System.out.println("Please enter Donation type:");
856         final String D_Type = sc.nextLine();
857         System.out.println("Please enter Campaign Name:");
858         final String Camp_Name = sc.nextLine();
859         System.out.println("Please enter Card Number:");
860         final int Check_Num = sc.nextInt();
861         sc.nextLine();
862         System.out.println("Please enter Card Type:");
863         final String CardType= sc.nextLine();
864         System.out.println("Please enter Expiry Date:");
865         final String date_E=sc.nextLine();
866         Date dateE=Date.valueOf(date_E);
867         try (final Connection connection = DriverManager.getConnection(URL)) {
868             try (final CallableStatement statement1 = connection.prepareCall(QUERY_TEMPLATE_17)) {
869                 {
870                     // Populate the query template with the data collected from the user
871                     statement1.setString(1,Org_Name );
872                     statement1.setDate(2, date);
873                     statement1.setFloat(3, D_amount);
874                     statement1.setString(4, D_Type);
875                     statement1.setString(5, Camp_Name);
876                     statement1.setInt(6, Check_Num);
877                     statement1.setString(7, CardType);
878                     statement1.setDate(8, dateE);
879
880
881                     System.out.println("Dispatching the query...");
882                     // Actually execute the populated query
883                     final int rows_inserted = statement1.executeUpdate();
884                     System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
885                 }
886             }
887         }
888     }
889
890     count5--;
891     break;
892
893     case "10": //Retrieve data of the client
894     try {
895         System.out.println("Please enter SSN:");
896         final int SSN = sc.nextInt();
897         //converting string into sql date
898         ResultSet resultSet=stmt.executeQuery("select Doctor_Name,Doctor_phone from Client WHERE SSN="+SSN+"");
899         System.out.println("Contents of the Client table:");
900         System.out.println("Doctor Name | Doctor Phone");
901         while (resultSet.next()) {
902             System.out.println(
903                 String.format("%s | %s ", resultSet.getString(1), resultSet.getInt(2)));
904         }
905     } catch (InputMismatchException e)
906     {
907         System.out.println("Caught Input mismatch");
908         sc.nextLine();
909         break;
910     }
911     catch (Exception e)
912     {
913         System.out.println("Couldn't execute the above query!");
914         System.out.println("Reason -- ");
915         System.out.println(e.getMessage());
916         System.out.println("Try again !!!");
917     }
918     break;
919
920     case "11": //Retrieve the data of Expenses
921     try {
922         System.out.println("Please enter Start Date:");
923         sc.nextLine();
924         final int date_Start=sc.nextInt();
925         System.out.println("Please enter End Date:");
926         final int date_End=sc.nextInt();

```

```

928      ResultSet resultSet1=stmt.executeQuery("select SSN,sum(Amount) from Expenses WHERE Expense_Date "
929      + "between "+date_Start+" and "+date_End+" group by SSN order by sum(Amount) asc");
930      System.out.println("Contents of the Expense table:");
931      System.out.println("SSN| Sum in total");
932      while (resultSet1.next()) {
933          System.out.println(
934              String.format("%s| %s ", resultSet1.getInt(1), resultSet1.getFloat(2)));
935      }
936      catch(InputMismatchException e)
937      {
938          System.out.println("Caught Input mismatch");
939          sc.nextLine();
940          break;
941      }
942      catch (Exception e)
943      {
944          System.out.println("Couldn't execute the above query!");
945          System.out.println("Reason -- ");
946          System.out.println(e.getMessage());
947          System.out.println("Try again !!!");
948      }
949      break;
950
951      case "12": //Retrieve list of Volunteers
952      try {
953          System.out.println("Please enter SSN:");
954          final int SSN = sc.nextInt();
955          ResultSet resultSet2=stmt.executeQuery("select SSN,Joining_Date,Training_Date,Training_Loc "
956          + " FROM Volunteers WHERE SSN in "
957          + "(SELECT C.SSN FROM Cares C,Serves S "
958          + "WHERE C.SSN="+SSN+" and S.SSN=C.SSN)");
959          System.out.println("Contents of the Volunteers table:");
960          System.out.println("SSN|Joining_Date|Training_Date|Training_Loc");
961          if (resultSet2!=null) {
962              while (resultSet2.next()) {
963                  System.out.println(
964                      String.format("%s| %s|%s|%s ", resultSet2.getInt(1), resultSet2.getDate(2),resultSet2.getDate(3),result
965                      ));
966              }

```

```

966      }
967      catch(InputMismatchException e)
968      {
969          System.out.println("Caught Input mismatch");
970          sc.nextLine();
971          break;
972      }
973      catch (Exception e)
974      {
975          System.out.println("Couldn't execute the above query!");
976          System.out.println("Reason -- ");
977          System.out.println(e.getMessage());
978          System.out.println("Try again !!!");
979      }
980      break;
981
982      case "13": //Retrieve info of client
983      try {
984          ResultSet resultSet2=stmt.executeQuery("SELECT Name,Mail_address,Email_address,Home_phone,"
985          + "Work_phone,Cell_phone from Person P,Client C "
986          + "WHERE C.SSN=P.SSN AND C.SSN IN (SELECT R.SSN FROM Cares R,Client C "
987          + "WHERE C.SSN=R.SSN AND R.T_Name IN (SELECT R.T_Name FROM Cares R,Teams T WHERE "
988          + "R.T_Name=T.T_Name AND T.T_Name IN (SELECT T.T_Name FROM Teams T,Sponsors S WHERE "
989          + "T.T_Name=S.T_Name AND S.Org_Name LIKE '[B-K]%' )))"
990          + " ORDER BY Name");
991          System.out.println("Contents of the Volunteers table:");
992          System.out.println("SSN|Joining_Date|Training_Date|Training_Loc");
993          while (resultSet2.next()) {
994              System.out.println(
995                  String.format("%s| %s|%s|%s|%s ", resultSet2.getString(1), resultSet2.getString(2),resultSet2.getStri
996                  ));
997          }
998          catch(InputMismatchException e)
999          {
1000              System.out.println("Caught Input mismatch");
1001              sc.nextLine();
1002              break;
1003          }
1004          catch (Exception e)
1005          {

```

```

973         catch (Exception e)
974         {
975             System.out.println("Couldn't execute the above query!");
976             System.out.println("Reason -- ");
977             System.out.println(e.getMessage());
978             System.out.println("Try again !!!");
979         }
980         break;
981
982     case "13": //Retrieve info of client
983     {
984         try {
985             ResultSet resultSet2=stmt.executeQuery("SELECT Name,Mail_address,Email_address,Home_phone,"
986             + "Work_phone,Cell_phone from Person P,Client C "
987             + "WHERE C.SSN=P.SSN AND C.SSN IN (SELECT R.SSN FROM Cares R,Client C "
988             + "WHERE C.SSN=R.SSN AND R.T_Name IN (SELECT R.T_Name FROM Cares R,Teams T WHERE "
989             + "R.T_Name=T.T_Name AND T.T_Name IN (SELECT T.T_Name FROM Teams T,Sponsors S WHERE "
990             + "T.T_Name=S.T_Name AND S.Org_Name LIKE '[B-K]%' )))"
991             + " ORDER BY Name");
992             System.out.println("Contents of the Volunteers table:");
993             System.out.println("SSN|Joining_Date|Training_Date|Training_Loc");
994             while (resultSet2.next()) {
995                 System.out.println(
996                     String.format("%s| %s| %s| %s| %s ", resultSet2.getString(1), resultSet2.getString(2),resultSet2.getStr
997                 ));
998             }
999             catch(InputMismatchException e)
1000             {
1001                 System.out.println("Caught Input mismatch");
1002                 sc.nextLine();
1003                 break;
1004             }
1005         }
1006         catch (Exception e)
1007         {
1008             System.out.println("Couldn't execute the above query!");
1009             System.out.println("Reason -- ");
1010             System.out.println(e.getMessage());
1011             System.out.println("Try again !!!");
1012         }
1013         break;

```

```

1012
1013     case "14": //Retrieve info of employees who are donors
1014     {
1015         try {
1016             ResultSet resultSet2=stmt.executeQuery("select distinct cc.SSN,Amount as Sum_amount,Anonymous from Person P,Dono
1017             + "WHERE CC.SSN=D.SSN AND E.SSN=CC.SSN UNION select distinct cc.SSN,CC_Amount as Sum_amount,Anonymous
1018             + "WHERE CC.SSN=D.SSN AND E.SSN=CC.SSN ");
1019             System.out.println("Contents of the table:");
1020             System.out.println("SSN|Sum_amount|Anonymous State");
1021             while (resultSet2.next()) {
1022                 System.out.println(
1023                     String.format("%s| %s| %s", resultSet2.getInt(1), resultSet2.getFloat(2),resultSet2.getString(3)));
1024             }
1025             catch(InputMismatchException e)
1026             {
1027                 System.out.println("Caught Input mismatch");
1028                 sc.nextLine();
1029                 break;
1030             }
1031         }
1032         catch (Exception e)
1033         {
1034             System.out.println("Couldn't execute the above query!");
1035             System.out.println("Reason -- ");
1036             System.out.println(e.getMessage());
1037             System.out.println("Try again !!!");
1038         }
1039         break;
1040
1041     case "15": //Retrieve info of teams
1042     {
1043         try {
1044             System.out.println("Please enter found Date:");
1045             final int date_Start=sc.nextInt();
1046             ResultSet resultSet1=stmt.executeQuery("select T_Name from Teams WHERE Date_T > "+date_Start+" ");
1047             System.out.println("Contents of the Team table:");
1048             System.out.println("Team Name");
1049             while (resultSet1.next()) {
1050                 System.out.println(
1051                     String.format("%s", resultSet1.getString(1)));
1052             }
1053         }

```

```

1049         catch(InputMismatchException e)
1050         {
1051             System.out.println("Caught Input mismatch");
1052             sc.nextLine();
1053             break;
1054         }
1055         catch (Exception e)
1056         {
1057             System.out.println("Couldn't execute the above query!");
1058             System.out.println("Reason -- ");
1059             System.out.println(e.getMessage());
1060             System.out.println("Try again !!!");
1061         }
1062         break;
1063     }
1064     case "16": //Increase salary
1065         int resultSet2 = stmt.executeUpdate("Update Employees SET E_Salary=E_Salary+0.1*E_Salary WHERE "
1066             + "SSN IN (SELECT E.SSN FROM Reports R,Employees E WHERE "
1067             + "E.SSN=R.SSN GROUP BY E.SSN HAVING COUNT(E.SSN)>1)");
1068         System.out.println(String.format("Done. %d rows inserted.", resultSet2));
1069         System.out.println("Connecting to the database...");
1070         // Get the database connection, create statement and execute it right away, as
1071         // no user input need be collected
1072         try (final Connection connection = DriverManager.getConnection(URL)) {
1073             System.out.println("Dispatching the query...");
1074             try (final Statement statement = connection.createStatement();
1075                 final ResultSet resultSet = statement.executeQuery(QUERY_TEMPLATE_18)) {
1076                 System.out.println("Contents of the Employee table:");
1077                 System.out.println("SSN | E_Salary | M_Status | HireDate ");
1078                 // Unpack the tuples returned by the database and print them out to the user
1079                 while (resultSet.next()) {
1080                     System.out.println(
1081                         String.format("%s | %s | %s | %s", resultSet.getInt(1), resultSet.getFloat(2),
1082                             resultSet.getString(3), resultSet.getDate(4)));
1083                 }
1084             }
1085         }
1086         break;
1087     }
1088     case "17": //Delete the client info
1089         int resultSet3 = stmt.executeUpdate("DELETE FROM Client WHERE "
1090             + "SSN IN (SELECT C.SSN FROM Client C,Needs N WHERE "
1091             + "Imp_Value<5 AND Type='transportation' ) AND SSN NOT IN"
1092             + " (SELECT SSN FROM Insurance_Policy )");
1093         System.out.println(String.format("Done. %d rows inserted.", resultSet3));
1094         System.out.println("Connecting to the database...");
1095         // Get the database connection, create statement and execute it right away, as
1096         // no user input need be collected
1097         try (final Connection connection = DriverManager.getConnection(URL)) {
1098             System.out.println("Dispatching the query...");
1099             try (final Statement statement = connection.createStatement();
1100                 final ResultSet resultSet = statement.executeQuery(QUERY_TEMPLATE_19)) {
1101                 System.out.println("Contents of the Client table:");
1102                 System.out.println("SSN | Doctor_Name | Attor_Name | Doctor_Phone |Attor_Phone| Date_assigned ");
1103                 // Unpack the tuples returned by the database and print them out to the user
1104                 while (resultSet.next()) {
1105                     System.out.println(
1106                         String.format("%s | %s | %s | %s| %s| %s", resultSet.getInt(1),resultSet.getString(2),resultSet.
1107                             resultSet.getInt(5), resultSet.getDate(6)));
1108                 }
1109             }
1110         }
1111         break;
1112     }
1113     case "18": //Import option
1114         try {
1115             System.out.println("Enter the source file name: ");
1116             final String sourcefile=br.readLine();
1117             FileInputStream fis = new FileInputStream("C:/Users/meghnath reddy/Desktop/DBMS Individual project1/" +sourcefile);
1118             DataInputStream dis = new DataInputStream(fis);
1119             BufferedReader br1 = new BufferedReader(new InputStreamReader(dis));
1120             String s;
1121             while((s = br1.readLine())!= null)
1122             {
1123                 String a[] = s.split(",");
1124                 stmt.executeUpdate("Insert into Teams values ('"+a[0]+ "','"+a[1]+'','"+a[2]+'')");
1125             }
1126         }

```

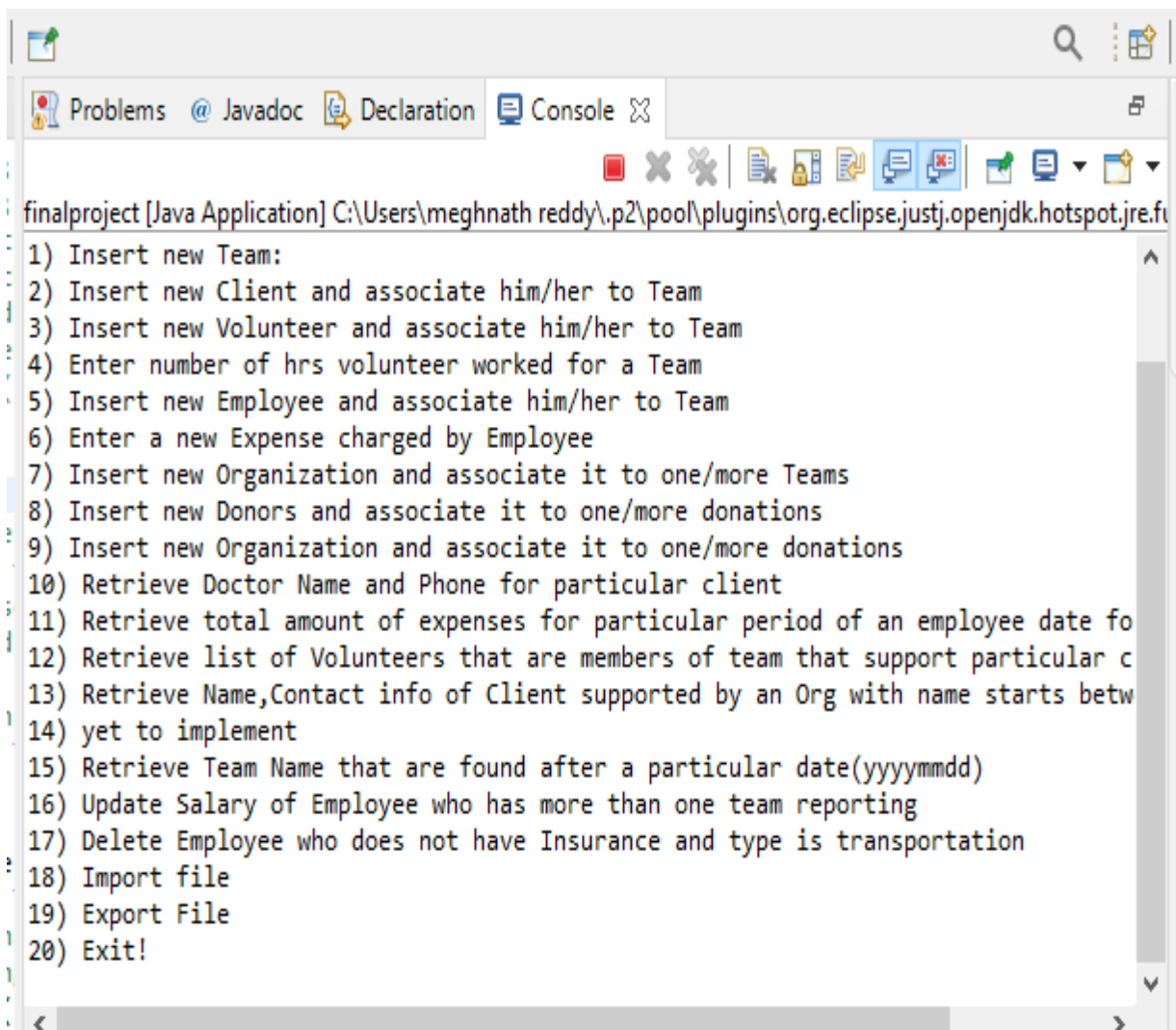


```

1126         System.out.println("Team/s has been added");
1127     }}
1128     catch(FileNotFoundException e) {
1129         e.printStackTrace();
1130     }
1131     catch(IOException ie){
1132         ie.printStackTrace();
1133     }
1134     break;
1135
1136     case "19": //Export option
1137     try {
1138         System.out.println("Enter the source destination file name: ");
1139         final String destfile=br.readLine();
1140         BufferedWriter bw = new BufferedWriter(new FileWriter("C:/Users/meghnath reddy/Desktop/DBMS Individual project1/"
1141         ResultSet res = stmt.executeQuery("select Name,Mail_address,Email_address from Person WHERE Mailing_list=0");
1142         while (res.next()) {
1143             bw.write(res.getString(1)+"\t");
1144             bw.write(res.getString(2)+"\t");
1145             bw.write(res.getString(3)+"\n");
1146         }bw.close();}
1147         catch(FileNotFoundException e) {
1148             e.printStackTrace();
1149         }
1150         catch(IOException ie){
1151             ie.printStackTrace();
1152         }
1153         break;
1154
1155     case "20": // Do nothing, the while loop will terminate upon the next iteration
1156         System.out.println("Exiting! Goodbuy!");
1157         break;
1158     default: // Unrecognized option, re-prompt the user for the correct one
1159         System.out.println(String.format("Unrecognized option: %s\n" + "Please try again!", option));
1160         break;
1161     }
1162 }
1163
1164 sc.close(); // Close the scanner before exiting the application

```





The screenshot shows the Eclipse IDE's console window. The title bar indicates the project is 'finalproject [Java Application]' located at 'C:\Users\meghnath reddy\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.fu'. The console contains a numbered list of 20 tasks:

- 1) Insert new Team:
- 2) Insert new Client and associate him/her to Team
- 3) Insert new Volunteer and associate him/her to Team
- 4) Enter number of hrs volunteer worked for a Team
- 5) Insert new Employee and associate him/her to Team
- 6) Enter a new Expense charged by Employee
- 7) Insert new Organization and associate it to one/more Teams
- 8) Insert new Donors and associate it to one/more donations
- 9) Insert new Organization and associate it to one/more donations
- 10) Retrieve Doctor Name and Phone for particular client
- 11) Retrieve total amount of expenses for particular period of an employee date fo
- 12) Retrieve list of Volunteers that are members of team that support particular c
- 13) Retrieve Name,Contact info of Client supported by an Org with name starts betw
- 14) yet to implement
- 15) Retrieve Team Name that are found after a particular date(yyyymmdd)
- 16) Update Salary of Employee who has more than one team reporting
- 17) Delete Employee who does not have Insurance and type is transportation
- 18) Import file
- 19) Export File
- 20) Exit!

## TASK 6

### 6.1) Screenshot showing successful execution of option 1

First add a person to the DB

```
finalproject [Java Application] C:\Users\meghnath reddy\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre
Welcome to the sample application!
Please enter SSN:
140
Please enter Person Name:
James
Please enter Birth Date:
2000-02-04
Please enter Race:
Latin
Please enter Gender:
Male
Please enter Profession:
Singer
Please enter Mail_address:
james@mail
Please enter Email_address:
james@email
Please enter Home_phone:
111
Please enter Work_phone:
112
```

```
finalproject [Java Application] C:\Users\meghnath reddy\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot
111
Please enter Work_phone:
112
Please enter Cell_phone:
113
Please enter Mailing_list state:
0
Dispatching the query...
Done. 1 rows inserted.

Please select one of the options below:
1) Insert new Team:
2) Insert new Client and associate him/her to Team
3) Insert new Volunteer and associate him/her to Team
4) Enter number of hrs volunteer worked for a Team
5) Insert new Employee and associate him/her to Team
6) Enter a new Expense charged by Employee
7) Insert new Organization and associate it to one/more Teams
8) Insert new Donors and associate it to one/more donations
9) Insert new Organization and associate it to one/more donations
10) Retrieve Doctor Name and Phone for particular client
```

## Second person to the DB

```
Please enter SSN:
141
Please enter Person Name:
Samuels
Please enter Birth Date:
1990-01-02
Please enter Race:
African
Please enter Gender:
Male
Please enter Profession:
Cricketer
Please enter Mail_address:
samuel@mail
Please enter Email_address:
samuel@email
Please enter Home_phone:
159
Please enter Work_phone:
160

mainproject [Java Application] C:\Users\megnn
Please enter Cell_phone:
161
Please enter Mailing_list state:
1
Dispatching the query...
Done. 1 rows inserted.
```

## Third person to the DB

```
Welcome to the sample application!
Please enter SSN:
143
Please enter Person Name:
Lop
Please enter Birth Date:
2156-03-01
Please enter Race:
LAT
Please enter Gender:
FEMALE
Please enter Profession:
DANCER
Please enter Mail_address:
lop@mail
Please enter Email_address:
lop@email
Please enter Home_phone:
145
Please enter Work_phone:
146
```

---

```

Please enter Cell_phone:
147
Please enter Mailing_list state:
0
Dispatching the query...
Done. 1 rows inserted.

```

#### Fourth person to the DB

---

```

Welcome to the sample application!
Please enter SSN:
145
Please enter Person Name:
rez
Please enter Birth Date:
1456-03-01
Please enter Race:
eng
Please enter Gender:
Female
Please enter Profession:
Buyer
Please enter Mail_address:
rez@mail
Please enter Email_address:
rez@email
Please enter Home_phone:
361
Please enter Work_phone:
362

Please enter Cell_phone:
363
Please enter Mailing_list state:
0
Dispatching the query...
Done. 1 rows inserted.

Please select one of the options b
1) Insert new Team:

```

After inserting 5 values the DB looks like:

	SSN	Name	Birthdate	Race	Gender	Profession	Mail_address	Email_address	Home_phone	Work_phone	Cell_phone	Mailing_list
1	140	James	2000-02-04	Latin	Male	Singer	james@mail	james@email	111	112	113	0
2	141	Samuels	1990-01-02	African	Male	Cricketer	samuel@mail	samuel@email	159	160	161	1
3	142	Ross	2000-02-01	Esp	Male	Skater	ross@mail	ross@email	231	232	233	1
4	143	Lop	2156-03-01	LAT	FEMALE	DANCER	lop@mail	lop@email	145	146	147	0
5	145	rez	1456-03-01	eng	Female	Buyer	rez@mail	rez@email	361	362	363	0

After selecting option-1:

1<sup>st</sup> team

```
1
Please enter Team Name:
IPL
Please enter Team Type:
CRIC
Please Date assigned:
20200101
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

2<sup>nd</sup> team:

```
1
Please enter Team Name:
RCB
Please enter Team Type:
POLISH
Please Date assigned:
20150201
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

3<sup>rd</sup> Team

```
1
Please enter Team Name:
lpu
Please enter Team Type:
college
Please Date assigned:
16670203
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

4<sup>th</sup> Team

```
1
Please enter Team Name:
klu
Please enter Team Type:
international
Please Date assigned:
20160403
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

## 5<sup>th</sup> Team

```
1
Please enter Team Name:
SAD
Please enter Team Type:
Emotion
Please Date assigned:
20140308
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

Check the DB to see if the 5 team rows are inserted

Results		Messages	
	T_Name	Type	Date_T
1	IPL	CRIC	20200101
2	klu	international	20160403
3	lpu	college	16670203
4	RCB	POLISH	20150201
5	SAD	Emotion	20140308

## 6.2) Screenshot showing successful execution of option 2

### 1<sup>ST</sup> CLIENT

```
2
Please enter SSN:
140
Please enter Doctor name:
Sam
Please enter Attor Name:
John
Please Doctor Phone:
147
Please enter Attor Phone:
148
Please Date assigned:
2020-01-15
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter SSN:
```

```
Number of teams to associate to
1
Please enter SSN:
140
Please enter Team Name to which you want to associate:
IPL
Dispatching the query...
Done. 1 rows inserted.
```

## 2<sup>ND</sup> Client

```
20) EXIT:
2
Please enter SSN:
141
Please enter Doctor name:
LJ
Please enter Attor Name:
KL
Please Doctor Phone:
356
Please enter Attor Phone:
357
Please Date assigned:
3001-02-04
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
2
Please enter SSN:
141
Please enter Team Name to which you want to associate:
RCB
Dispatching the query...
Done. 1 rows inserted.
Please enter SSN:
141
Please enter Team Name to which you want to associate:
SAD
Dispatching the query...
Done. 1 rows inserted.

Please select one of the options below:
```

### 3<sup>RD</sup> CLIENT:

```
; 2
; Please enter SSN:
t 143
t Please enter Doctor name:
d lpq
e Please enter Attor Name:
( qwe
  Please Doctor Phone:
  369
  Please enter Attor Phone:
e 370
  Please Date assigned:
s 2016-02-04
d Connecting to the database...
  Dispatching the query...
n Done. 1 rows inserted.
  Number of teams to associate to
  1
```

```
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter SSN:
143
Please enter Team Name to which you want to associate:
SAD
Dispatching the query...
Done. 1 rows inserted.
```

### 4<sup>TH</sup> Client

```
2
Please enter SSN:
142
Please enter Doctor name:
KM
Please enter Attor Name:
MP
Please Doctor Phone:
369
Please enter Attor Phone:
370
Please Date assigned:
1222-02-02
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter SSN:
142
Please enter Team Name to which you want to associate:
klu
Dispatching the query...
Done. 1 rows inserted.
```



## 5<sup>th</sup> Client

```
20/ EX10:  
2  
Please enter SSN:  
145  
Please enter Doctor name:  
NM  
Please enter Attor Name:  
MNP  
Please Doctor Phone:  
136  
Please enter Attor Phone:  
168  
Please Date assigned:  
3129-03-01  
Connecting to the database...  
Dispatching the query...  
Done. 1 rows inserted.  
Number of teams to associate to  
1  
Please enter SSN:  
145  
Please enter Team Name to which you want to associate:  
IPL  
Dispatching the query...  
Done. 1 rows inserted.
```

## Check DB if clients are inserted

Results		Messages				
	SSN	Doctor_Name	Attor_Name	Doctor_phone	Attor_phone	Date_assigned
1	140	Sam	John	147	148	2020-01-15
2	141	LJ	KL	356	357	3001-02-04
3	142	KM	MP	369	370	1222-02-02
4	143	lpq	qwe	369	370	2016-02-04
5	145	NM	MNP	136	168	3129-03-01

### 6.3) Screenshot showing successful execution of option 3

#### 1<sup>st</sup> Volunteer

##### Option3) Input

```
3
Please enter SSN:
141
Please enter Joining Date:
2019-02-03
Please Training Date:
2019-03-05
Please enter Training Loc :
HYD
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter SSN:
141
Please enter Team Name to which you want to associate:
IPL
Please enter HOURS:
5

Please enter Specific month:
JAN
Dispatching the query...
Done. 1 rows inserted.
```

#### 2<sup>nd</sup>

```
3
Please enter SSN:
140
Please enter Joining Date:
2301-02-04
Please Training Date:
2365-01-02
Please enter Training Loc :
lop
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter SSN:
140
Please enter Team Name to which you want to associate:
IPL
Please enter HOURS:
5
Please enter Specific month:
DEC
Dispatching the query...
Done. 1 rows inserted.
```

3<sup>rd</sup> :

```
3
Please enter SSN:
142
Please enter Joining Date:
1444-05-06
Please Training Date:
1236-05-05
Please enter Training Loc :
kop
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter SSN:
142
Please enter Team Name to which you want to associate:
SAD
Please enter HOURS:
45
Please enter Specific month:
JUN
Dispatching the query...
```

4<sup>TH</sup>

```
3
Please enter SSN:
145
Please enter Joining Date:
1444-02-01
Please Training Date:
1456-03-30
Please enter Training Loc :
lopk
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter SSN:
145
Please enter Team Name to which you want to associate:
RCB
Please enter HOURS:
12
Please enter Specific month:
APR
Dispatching the query...
Done. 1 rows inserted.
```

5<sup>TH</sup>:

```
20) exit:
3
Please enter SSN:
143
Please enter Joining Date:
1269-03-04
Please Training Date:
2356-04-08
Please enter Training Loc :
NMD
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter SSN:
143
Please enter Team Name to which you want to associate:
SAD
Please enter HOURS:
12
Please enter Specific month:
JUN
Dispatching the query...
Done. 1 rows inserted.
```

Check DB if insertion is done:

Results		Messages		
	SSN	Joining_Date	Training_Date	Training_Loc
1	140	2301-02-04	2365-01-02	lop
2	141	2019-02-03	2019-03-05	HYD
3	142	1444-05-06	1236-05-05	kop
4	143	1269-03-04	2356-04-08	NMD
5	145	1444-02-01	1456-03-30	lopk

#### 6.4) Screenshot showing successful execution of option 4

1<sup>ST</sup>:

```
4
Please enter SSN:
141
Please enter Team Name to which you want to associate:
IPL
Please enter HOURS:
20
Please enter Specific month:
FEB
Dispatching the query...
Done. 1 rows inserted.
```

2<sup>ND</sup>:

```
4
Please enter SSN:
140
Please enter Team Name to which you want to associate:
RCB
Please enter HOURS:
15
Please enter Specific month:
FEB
Dispatching the query...
Done. 1 rows inserted.
```

3<sup>RD</sup>:

```
4
Please enter SSN:
142
Please enter Team Name to which you want to associate:
SAD
Please enter HOURS:
36
Please enter Specific month:
APR
Dispatching the query...
Done. 1 rows inserted.
```

4<sup>TH</sup>:

```
4
Please enter SSN:
143
Please enter Team Name to which you want to associate:
RCB
Please enter HOURS:
23
Please enter Specific month:
JUN
Dispatching the query...
Done. 1 rows inserted.
```

5<sup>TH</sup>:

```
4
Please enter SSN:
145
Please enter Team Name to which you want to associate:
klu
Please enter HOURS:
15
Please enter Specific month:
AUG
Dispatching the query...
Done. 1 rows inserted.
```

Check if insertion is successful:

	SSN	T_Name	Hours	Active_State	Specific_Month
1	141	IPL	5	0	JAN
2	141	IPL	20	0	FEB
3	140	IPL	5	0	DEC
4	142	SAD	45	0	JUN
5	145	RCB	12	0	APR
6	143	SAD	12	0	JUN
7	142	SAD	36	0	APR
8	140	RCB	15	0	FEB
9	143	RCB	23	0	JUN
10	145	klu	15	0	AUG

## 6.5) Screenshot showing successful execution of option 5

1<sup>st</sup>:

```
SQL> EXEC  
5  
Please enter SSN:  
140  
Please enter Salary of Employee:  
1500  
Please enter Marital Status:  
Yes  
Please enter Hire Date:  
2020-01-02  
Connecting to the database...  
Dispatching the query...  
Done. 1 rows inserted.  
Number of teams to associate to  
1  
Please enter SSN:  
140  
Please enter Team Name to which you want to associate:  
IPL  
Please enter Report Date:  
2020-03-05  
Please enter Report Description:  
Rent  
Dispatching the query...  
Done. 1 rows inserted.
```

2<sup>nd</sup>:

```
SQL> EXEC  
5  
Please enter SSN:  
141  
Please enter Salary of Employee:  
1200  
Please enter Marital Status:  
No  
Please enter Hire Date:  
2001-02-03  
Connecting to the database...  
Dispatching the query...  
Done. 1 rows inserted.  
Number of teams to associate to  
2  
Please enter SSN:  
141  
Please enter Team Name to which you want to associate:  
SAD  
Please enter Report Date:  
2001-03-05  
Please enter Report Description:  
Access  
Dispatching the query...  
Done. 1 rows inserted.  
Please enter SSN:  
141  
Please enter Team Name to which you want to associate:  
RCB  
Please enter Report Date:  
2150-03-04  
Please enter Report Description:  
Break  
Dispatching the query...  
Done. 1 rows inserted.  
<
```

3<sup>RD</sup>:

```
5
Please enter SSN:
142
Please enter Salary of Employee:
3000
Please enter Marital Status:
Yes
Please enter Hire Date:
2016-03-04
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter SSN:
142
Please enter Team Name to which you want to associate:
IPL
Please enter Report Date:
2016-02-05
Please enter Report Description:
Satisfied
Dispatching the query...
Done. 1 rows inserted.
```

4<sup>th</sup>:

```
5
Please enter SSN:
143
Please enter Salary of Employee:
5000
Please enter Marital Status:
Yes
Please enter Hire Date:
2018-03-05
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
2
Please enter SSN:
143
Please enter Team Name to which you want to associate:
RCB
Please enter Report Date:
2013-04-03
Please enter Report Description:
Network
Dispatching the query...
Done. 1 rows inserted.
Please enter SSN:
143
Please enter Team Name to which you want to associate:
SAD
Please enter Report Date:
2017-06-08
Please enter Report Description:
Lack
Dispatching the query...
Done. 1 rows inserted.
```



5<sup>TH</sup>:

```
SQL> EXEC:
5
Please enter SSN:
145
Please enter Salary of Employee:
2000
Please enter Marital Status:
Yes
Please enter Hire Date:
2013-05-04
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter SSN:
145
Please enter Team Name to which you want to associate:
RCB
Please enter Report Date:
2016-05-04
Please enter Report Description:
Risk
Dispatching the query...
Done. 1 rows inserted.
```

Employess Table:

Results		Messages		
	SSN	E_Salary	M_Status	HireDate
1	140	1500	Yes	2020-01-02
2	141	1200	No	2001-02-03
3	142	3000	Yes	2016-03-04
4	143	5000	Yes	2018-03-05
5	145	2000	Yes	2013-05-04

Reports Table:

Results		Messages		
	SSN	T_Name	R_DATE	Desc_1
1	140	IPL	2020-03-05	Rent
2	142	IPL	2016-02-05	Satisfied
3	141	RCB	2150-03-04	Break
4	143	RCB	2013-04-03	Network
5	145	RCB	2016-05-04	Risk
6	141	SAD	2001-03-05	Access
7	143	SAD	2017-06-08	Lack

## 6.6) Screenshot showing successful execution of option 6

1<sup>st</sup>:

```
6
Please enter SSN:
140
Please enter Expense Amount:
4500
Please enter Expense Description:
Laptop
Please enter Expense Date:
20160403
Dispatching the query...
Done. 1 rows inserted.
```

2<sup>nd</sup>:

```
6
Please enter SSN:
141
Please enter Expense Amount:
4500
Please enter Expense Description:
PC
Please enter Expense Date:
20160409
Dispatching the query...
Done. 1 rows inserted.
```

3<sup>rd</sup>:

```
6
Please enter SSN:
142
Please enter Expense Amount:
7800
Please enter Expense Description:
PC
Please enter Expense Date:
20160103
Dispatching the query...
Done. 1 rows inserted.
```

4<sup>th</sup>:

```
6
Please enter SSN:
143
Please enter Expense Amount:
2300
Please enter Expense Description:
Gold
Please enter Expense Date:
20200506
Dispatching the query...
Done. 1 rows inserted.
```

5<sup>th</sup>:

```
6
Please enter SSN:
145
Please enter Expense Amount:
2300
Please enter Expense Description:
LPG
Please enter Expense Date:
20160809
Dispatching the query...
Done. 1 rows inserted.
```

#### EXPENSES TABLE:

Results		Messages		
	SSN	Amount	Desc_1	Expense_Date
1	142	7800	PC	20160103
2	140	4500	Laptop	20160403
3	141	4500	PC	20160409
4	145	2300	LPG	20160809
5	143	2300	Gold	20200506

#### 6.7) Screenshot showing successful execution of option 7

1<sup>st</sup>:

```
7
Please enter Organization Name:
Warrior
Please enter Organization Address:
BAN
Please Org Number:
124
Please enter Org Contact :
Ben
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter Team Name to which you want to associate:
RCB
Please enter Organization Name:
Warrior
Dispatching the query...
Done. 1 rows inserted.
```

2<sup>nd</sup>:

```
7
Please enter Organization Name:
Pink
Please enter Organization Address:
Aus
Please Org Number:
145
Please enter Org Contact :
Steve
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
2
Please enter Team Name to which you want to associate:
SAD
Please enter Organization Name:
Pink
Dispatching the query...
Done. 1 rows inserted.
Please enter Team Name to which you want to associate:
RCB
Please enter Organization Name:
Pink
Dispatching the query...
Done. 1 rows inserted.
```

3<sup>rd</sup>:

```
7
Please enter Organization Name:
Help
Please enter Organization Address:
Ind
Please Org Number:
199
Please enter Org Contact :
Ram
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter Team Name to which you want to associate:
SAD
Please enter Organization Name:
Help
Dispatching the query...
Done. 1 rows inserted.
```

4<sup>th</sup>:

```
7
Please enter Organization Name:
Hero
Please enter Organization Address:
Pak
Please Org Number:
143
Please enter Org Contact :
Tim
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter Team Name to which you want to associate:
klu
Please enter Organization Name:
Hero
Dispatching the query...
Done. 1 rows inserted.
```

5<sup>th</sup>:

```
7
Please enter Organization Name:
Kings
Please enter Organization Address:
Zim
Please Org Number:
236
Please enter Org Contact :
sTEVE
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of teams to associate to
1
Please enter Team Name to which you want to associate:
lpu
Please enter Organization Name:
Kings
Dispatching the query...
Done. 1 rows inserted.
```

Organization Table:

Results		Messages		
	Org_Name	Org_address	Org_Number	Org_Contact
1	Help	Ind	199	Ram
2	Hero	Pak	143	Tim
3	Kings	Zim	236	sTEVE
4	Pink	Aus	145	Steve
5	Warrior	BAN	124	Ben

## 6.8) Screenshot showing successful execution of option 8

1<sup>st</sup>:

```
8
Please enter SSN:
140
Please enter Anonymous state of Donor :
No
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of donations to associate to
1
Please enter Donation payment 1:check and 2:Credit:
1
Please enter SSN:
140
Please enter Date of Donation:
2015-03-04
Please enter Donation Amount:
2000
Please enter Donation type:
Self
Please enter Campaign Name:
Wonder
Please enter Cheque Number:
2150
Dispatching the query...
Done. 1 rows inserted.
```

2<sup>nd</sup>:

```
8
Please enter SSN:
141
Please enter Anonymous state of Donor :
No
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of donations to associate to
1
Please enter Donation payment 1:check and 2:Credit:
2
Please enter SSN:
141
Please enter Date of Donation:
2013-01-06
Please enter Donation Amount:
2300
Please enter Donation type:
Record
Please enter Campaign Name:
Revolution
Please enter Card Number:
12364
Please enter Card Type:
Own
Please enter Expiry Date:
2016-03-04
Dispatching the query...
Done. 1 rows inserted.
```

3<sup>rd</sup>:

```
8
Please enter SSN:
142
Please enter Anonymous state of Donor :
Yes
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of donations to associate to
2
Please enter Donation payment 1:check and 2:Credit:
1
Please enter SSN:
142
Please enter Date of Donation:
2016-02-09
Please enter Donation Amount:
451
Please enter Donation type:
Valid
Please enter Campaign Name:
lp
Please enter Cheque Number:
2301
Dispatching the query...
Done. 1 rows inserted.
Please enter Donation payment 1:check and 2:Credit:
1
Please enter SSN:
142
Please enter Date of Donation:
2013-09-11
```

```
Please enter Donation Amount:
2600
Please enter Donation type:
Pin
Please enter Campaign Name:
klp
Please enter Cheque Number:
12365
Dispatching the query...
Done. 1 rows inserted.
```

4<sup>th</sup>:

```
8
Please enter SSN:
143
Please enter Anonymous state of Donor :
No
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of donations to associate to
1
Please enter Donation payment 1:check and 2:Credit:
2
Please enter SSN:
143
Please enter Date of Donation:
2016-04-07
Please enter Donation Amount:
2300
Please enter Donation type:
Care
Please enter Campaign Name:
kick
Please enter Card Number:
14593
Please enter Card Type:
simple
Please enter Expiry Date:
2013-06-04
Dispatching the query...
Done. 1 rows inserted.
```

5<sup>th</sup>:

```
8
Please enter SSN:
145
Please enter Anonymous state of Donor :
Yes
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of donations to associate to
1
Please enter Donation payment 1:check and 2:Credit:
1
Please enter SSN:
145
Please enter Date of Donation:
2019-04-07
Please enter Donation Amount:
1200
Please enter Donation type:
Pil
Please enter Campaign Name:
Sink
Please enter Cheque Number:
1456
Dispatching the query...
Done. 1 rows inserted.
```

External\_Org Table after insetion:

Results		Messages		
	Org_Name	Org_address	Org_Number	Org_Contact
1	Help	Ind	199	Ram
2	Hero	Pak	143	Tim
3	Kings	Zim	236	sTEVE
4	Pink	Aus	145	Steve
5	Warrior	BAN	124	Ben

Donors Table after Insertion:

Results		Messages
	SSN	Anonymous
1	140	No
2	141	No
3	142	Yes
4	143	No
5	145	Yes



### Check Table:

Results		Messages				
	SSN	C_Date	Amount	D_Type	Campaign_Name	Cheque_Number
1	140	2015-03-04	2000	Self	Wonder	2150
2	142	2013-09-11	2600	Pin	klp	12365
3	142	2016-02-09	451	Valid	lp	2301
4	145	2019-04-07	1200	Pil	Sink	1456

### Credit Card:

Results		Messages						
	SSN	CC_Date	CC_Amount	D_Type	Campaign_Name	Card_Number	Card_Type	Exp_Date
1	141	2013-01-06	2300	Record	Revolution	12364	Own	2016-03-04
2	143	2016-04-07	2300	Care	kick	14593	simple	2013-06-04

### 6.9) Screenshot showing successful execution of option 9

1<sup>st</sup>:

```
9
Please enter Organization Name:
Pls
Please enter Organization Address:
zim
Please Org Number:
1450
Please enter Org Contact :
Sim
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Please enter Organization Name:
Pls
Please enter Anonymous state of Donor :
Yes
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of donations to associate to
1
Please enter Donation payment 1:check and 2:Credit:
1
Please enter Org Name:
Pls
Please enter Date of Donation:
2016-04-03
Please enter Donation Amount:
14500
Please enter Donation type:
Simple
Please enter Campaign Name:
Care
```

```
Care
Please enter Cheque Number:
1459
Dispatching the query...
Done. 1 rows inserted.
```

2<sup>nd</sup>:

```
9
Please enter Organization Name:
Lpp
Please enter Organization Address:
Ind
Please Org Number:
2630
Please enter Org Contact :
Sam
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Please enter Organization Name:
Lpp
Please enter Anonymous state of Donor :
No
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of donations to associate to
1
Please enter Donation payment 1:check and 2:Credit:
2
Please enter Org Name:
Lpp
Please enter Date of Donation:
2013-04-06
Please enter Donation Amount:
200
Please enter Donation type:
Dare
Please enter Campaign Name:
P1

Please enter Card Number:
14560
Please enter Card Type:
Credit
Please enter Expiry Date:
2019-06-07
Dispatching the query...
Done. 1 rows inserted.
```

3<sup>rd</sup>:

```
9
Please enter Organization Name:
Pol
Please enter Organization Address:
Pak
Please Org Number:
14590
Please enter Org Contact :
Kim
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Please enter Organization Name:
Pol
Please enter Anonymous state of Donor :
No
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of donations to associate to
1
Please enter Donation payment 1:check and 2:Credit:
1
Please enter Org Name:
Pol
Please enter Date of Donation:
2013-04-08
Please enter Donation Amount:
1500
Please enter Donation type:
P1
Please enter Campaign Name:
Run
Please enter Cheque Number:
14560
Dispatching the query...
Done. 1 rows inserted.
```

4<sup>th</sup>:

```
9
Please enter Organization Name:
UPS
Please enter Organization Address:
Aus
Please Org Number:
1470
Please enter Org Contact :
Pem
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Please enter Organization Name:
UPS
Please enter Anonymous state of Donor :
No
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of donations to associate to
1
Please enter Donation payment 1:check and 2:Credit:
2
Please enter Org Name:
UPS
Please enter Date of Donation:
2016-07-03
Please enter Donation Amount:
1500
Please enter Donation type:
Self
Please enter Campaign Name:
lop
```

Please enter Card Number:  
14593  
Please enter Card Type:  
Savings  
Please enter Expiry Date:  
2019-03-07  
Dispatching the query...  
Done. 1 rows inserted.

5<sup>th</sup>:

```
9
Please enter Organization Name:
Tall
Please enter Organization Address:
US
Please Org Number:
1236
Please enter Org Contact :
STEVE
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Please enter Organization Name:
Tall
Please enter Anonymous state of Donor :
No
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
Number of donations to associate to
1
Please enter Donation payment 1:check and 2:Credit:
1
Please enter Org Name:
Tall
Please enter Date of Donation:
2016-08-06
Please enter Donation Amount:
2300
Please enter Donation type:
Lap
Please enter Campaign Name:
Mon

Please enter Cheque Number:
2360
Dispatching the query...
Done. 1 rows inserted.
```

## Tables after insertion:

Results		Messages		
	Org_Name	Org_address	Org_Number	Org_Contact
1	Help	Ind	199	Ram
2	Hero	Pak	143	Tim
3	Kings	Zim	236	sTEVE
4	Lpp	Ind	2630	Sam
5	Pink	Aus	145	Steve
6	Pls	zim	1450	Sim
7	Pol	Pak	14590	Kim
8	Tall	US	1236	STEVE
9	UPS	Aus	1470	Pem
10	Warrior	BAN	124	Ben

Results		Messages	
	Org_Name	Anonymous	
1	Lpp	No	
2	Pls	Yes	
3	Pol	No	
4	Tall	No	
5	UPS	No	

	Org_Name	Ex_Date	Ex_Amount	Ex_Type	Campaign_Name	Cheque_Number
1	Pls	2016-04-03	14500	Simple	Care	1459
2	Pol	2013-04-08	1500	P1	Run	14560
3	Tall	2016-08-06	2300	Lap	Mon	2360

Results		Messages						
	Org_Name	ExCC_Date	ExCC_Amount	ExCC_Type	ExCCCampaign_Name	ExCard_number	ExCard_Type	Ex_Exp_Date
1	Lpp	2013-04-06	200	Dare	P1	14560	Credit	2019-06-07
2	UPS	2016-07-03	1500	Self	lop	14593	Savings	2019-03-07

## 6.10) Screenshot showing successful execution of option 10

1<sup>st</sup>:

```
20) Exit!  
10  
Please enter SSN:  
140  
Contents of the Client table:  
Doctor Name | Doctor Phone  
Sam | 147
```

2<sup>nd</sup>:

```
20) Exit!  
10  
Please enter SSN:  
141  
Contents of the Client table:  
Doctor Name | Doctor Phone  
LJ | 356
```

3<sup>rd</sup>:

```
20) Exit!  
10  
Please enter SSN:  
142  
Contents of the Client table:  
Doctor Name | Doctor Phone  
KM | 369
```

4<sup>th</sup>:

```
20) Exit!  
10  
Please enter SSN:  
143  
Contents of the Client table:  
Doctor Name | Doctor Phone  
lpq | 369
```

## 6.11) Screenshot showing successful execution of option 11

1<sup>st</sup>:

```
11
Please enter Start Date:
20160101
Please enter End Date:
20160431
Contents of the Expense table:
SSN| Sum in total
140| 4500.0
141| 4500.0
142| 7800.0
```

2<sup>nd</sup>:

```
11
Please enter Start Date:
20160501
Please enter End Date:
20160901
Contents of the Expense table:
SSN| Sum in total
145| 2300.0
```

## 6.12) Screenshot showing successful execution of option 12

1<sup>st</sup>:

```
12
Please enter SSN:
140
Contents of the Volunteers table:
SSN|Joining_Date|Training_Date|Training_Loc
140| 2301-02-04|2365-01-02|lop
```

2<sup>ND</sup>:

```
12
Please enter SSN:
142
Contents of the Volunteers table:
SSN|Joining_Date|Training_Date|Training_Loc
142| 1444-05-06|1236-05-05|kop
```

6.13) Screenshot showing successful execution of option 13

```
20) exit:
13
Contents of the Volunteers table:
SSN|Joining_Date|Training_Date|Training_Loc
Lop| lop@mail|lop@email|145|146|147
Ross| ross@mail|ross@email|231|232|233
Samuels| samuel@mail|samuel@email|159|160|161
```

6.14) Screenshot showing successful execution of option 14

```
14
Contents of the table:
SSN|Sum_amount|Anonymous State
140| 2000.0|No
141| 2300.0|No
142| 451.0|Yes
142| 2600.0|Yes
143| 2300.0|No
145| 1200.0|Yes
```

6.15) Screenshot showing successful execution of option 15

```
15
Please enter found Date:
20150101
Contents of the Team table:
Team Name
RCB
klu
IPL
```

6.16) Screenshot showing successful execution of option 16

```
16
Done. 2 rows inserted.
Connecting to the database...
Dispatching the query...
Contents of the Employee table:
SSN | E_Salary | M_Status | HireDate
140 | 1500.0 | Yes | 2020-01-02
141 | 1320.0 | No | 2001-02-03
142 | 3000.0 | Yes | 2016-03-04
143 | 5500.0 | Yes | 2018-03-05
145 | 2000.0 | Yes | 2013-05-04
```

6.17) Screenshot showing successful execution of option 17

No change



```

17
Done. 0 rows inserted.
Connecting to the database...
Dispatching the query...
Contents of the Client table:
SSN | Doctor_Name | Attor_Name | Doctor_Phone | Attor_Phone | Date_assigned
140 | Sam | John | 147|148|2020-01-15
141 | LJ | KL | 356|357|3001-02-04
142 | KM | MP | 369|370|1222-02-02
143 | lpq | qwe | 369|370|2016-02-04
145 | NM | MNP | 136|168|3129-03-01

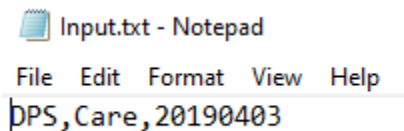
```

6.18) Screenshot showing successful execution of option 18

Before insertion Team table looks like:

Results		Messages	
	T_Name	Type	Date_T
1	IPL	CRIC	20200101
2	klu	international	20160403
3	lpu	college	16670203
4	RCB	POLISH	20150201
5	SAD	Emotion	20140308

Input file in .txt format



Taking input file name from the User:

```

18
Enter the source file name:
Input.txt
One Team has been added

```

Team table after insertion:

## Results Messages

	T_Name	Type	Date_T
1	DPS	Care	20190403
2	IPL	CRIC	20200101
3	klu	international	20160403
4	lpu	college	16670203
5	RCB	POLISH	20150201
6	SAD	Emotion	20140308

## 6.19) Screenshot showing successful execution of option 19

Person table :

	SSN	Name	Birthdate	Race	Gender	Profession	Mail_address	Email_address	Home_phone	Work_phone	Cell_phone	Mailing_list
1	140	James	2000-02-04	Latin	Male	Singer	james@mail	james@email	111	112	113	0
2	141	Samuels	1990-01-02	African	Male	Cricketer	samuel@mail	samuel@email	159	160	161	1
3	142	Ross	2000-02-01	Esp	Male	Skater	ross@mail	ross@email	231	232	233	1
4	143	Lop	2156-03-01	LAT	FEMALE	DANCER	lop@mail	lop@email	145	146	147	0
5	145	rez	1456-03-01	eng	Female	Buyer	rez@mail	rez@email	361	362	363	0

Taking input file name from the User:

```
20) EXIT:
19
Enter the source destination file name:
Output.txt
```

After export the .txt has new values:

```
Output.txt - Notepad
File Edit Format View Help
James james@mail james@email
Lop lop@mail lop@email
rez rez@mail rez@email
```

## 6.20) Screenshot showing successful execution of option 20

**Exit option:**

```
20) Exit!  
20  
Exiting! Goodbuy!
```

**If user selects option which is not in 1-20:**

```
25  
Unrecognized option: 25  
Please try again!
```

## 6.21) Screenshots to demonstrate Error handling

- 1) When the data type does not match the defined SQL table then we have Input mismatch exception.

```
Connecting to the database...  
Dispatching the query...  
Done. 1 rows inserted.  
Number of teams to associate to  
1  
Please enter SSN:  
RCB  
Exception in thread "main" java.util.InputMismatchException  
    at java.base/java.util.Scanner.throwFor(Scanner.java:939)  
    at java.base/java.util.Scanner.next(Scanner.java:1594)  
    at java.base/java.util.Scanner.nextInt(Scanner.java:2258)  
    at java.base/java.util.Scanner.nextInt(Scanner.java:2212)  
    at finalproject.main(finalproject.java:233)
```

- 2) When user tries to access the Team name which is not present in the Team table we get foreign key violation

```
1
Please enter SSN:
145
Please enter Team Name to which you want to associate:
CRIC
Dispatching the query...
Exception in thread "main" com.microsoft.sqlserver.jdbc.SQLServerException: The INSERT statement conflicted with the FOREIGN KEY constraint "FK_Cares_T_Name__77B5A9F0". The
at com.microsoft.sqlserver.jdbc@8.4.1.jre14/com.microsoft.sqlserver.jdbc.SQLServerException.makeFromDatabaseError(SQLServerException.java:262)
at com.microsoft.sqlserver.jdbc@8.4.1.jre14/com.microsoft.sqlserver.jdbc.SQLServerStatement.getNextResult(SQLServerStatement.java:1632)
at com.microsoft.sqlserver.jdbc@8.4.1.jre14/com.microsoft.sqlserver.jdbc.SQLServerPreparedStatement.doExecutePreparedStatement(SQLServerPreparedStatement.java:602)
at com.microsoft.sqlserver.jdbc@8.4.1.jre14/com.microsoft.sqlserver.jdbc.SQLServerPreparedStatement$PrepStmtExecCmd.doExecute(SQLServerPreparedStatement.java:524)
at com.microsoft.sqlserver.jdbc@8.4.1.jre14/com.microsoft.sqlserver.jdbc.TDSCommand.execute(IOBuffer.java:7375)
at com.microsoft.sqlserver.jdbc@8.4.1.jre14/com.microsoft.sqlserver.jdbc.SQLServerConnection.executeCommand(SQLServerConnection.java:3206)
at com.microsoft.sqlserver.jdbc@8.4.1.jre14/com.microsoft.sqlserver.jdbc.SQLServerStatement.executeCommand(SQLServerStatement.java:247)
at com.microsoft.sqlserver.jdbc@8.4.1.jre14/com.microsoft.sqlserver.jdbc.SQLServerStatement.executeStatement(SQLServerStatement.java:222)
at com.microsoft.sqlserver.jdbc@8.4.1.jre14/com.microsoft.sqlserver.jdbc.SQLServerPreparedStatement.executeUpdate(SQLServerPreparedStatement.java:473)
at finalproject.main(finalproject.java:247)
```

- 3) When we try to insert a same Team\_name into Teams table then primary key violation error is displayed.

```
1
Please enter Team Name:
RCB
Please enter Team Type:
PLS
Please Date assigned in format YYYYMMDD:
20120304
Connecting to the database...
Dispatching the query...
Couldn't execute the above query!
Reason --
Violation of PRIMARY KEY constraint 'PK_Teams__C3AF9CB6B239BDF'. Cannot insert duplicate key in object 'dbo.Teams'.
Try again !!!
```

## TASK 7

### 7.1. Web database application source program and screenshots showing Its successful compilation

#### 7.1.1 Data handlers file

```
1 package client_db;
2
3
4 import java.sql.Connection;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8 import java.sql.DriverManager;
9 import java.sql.PreparedStatement;
10 public class DataHandler {
11     private Connection conn;
12     // Azure SQL connection credentials
13     private static String server = "chal0018-sql-server.database.windows.net";
14     private static String database = "HW-2";
15     private static String username = "chal0018";
16     private static String password = "Sridevi$123";
17     // Resulting connection string
18     final static String URL = String.format(
19         "jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;trustServerCertificate=false;hostNameInCertificate="
20         server, database, username, password);
21     // Initialize and save the database connection
22     private void getDBConnection() throws SQLException {
23         if (conn != null) {
24             return;
25         }
26         this.conn = DriverManager.getConnection(URL);
27     }
28     // Return the result of selecting everything from the movie_night table
29     public ResultSet getAllMovies() throws SQLException {
30         getDBConnection();
31
32         final String sqlQuery = "SELECT * FROM Teams;";
33         final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
34         return stmt.executeQuery();
35
36
37     public boolean addMovie1(int Date_T) throws SQLException {
38         getDBConnection(); // Prepare the database connection
39         // Prepare the SQL statement
40         Connection conn = DriverManager.getConnection(URL); //establish connection
41         Statement stmt = conn.createStatement(); //to execute sql queries
42         ResultSet resultSet1=stmt.executeQuery("select T_Name from Teams WHERE Date_T > "+Date_T+" ");
43         // Replace the '?' in the above statement with the given attribute values
44         while (resultSet1.next()) {
45             System.out.println(
46                 String.format("%s", resultSet1.getString(1)));
47         }
48         return true;}
49         // Execute the query, if only one record is updated, then we indicate success by returning true
50
51
52
53
54     // Inserts a record into the movie_night table with the given attribute values
55     public boolean addMovie(
56         String T_Name, String Type, int Date_T) throws SQLException {
57         getDBConnection(); // Prepare the database connection
58         // Prepare the SQL statement
59         final String sqlQuery =
60             "INSERT INTO Teams " +
61             "(T_Name, Type, Date_T) " +
62             "VALUES " +
63             "(?, ?, ?)";
64         final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
65         // Replace the '?' in the above statement with the given attribute values
66         stmt.setString(1, T_Name);
67         stmt.setString(2, Type);
68         stmt.setInt(3, Date_T);
69         // Execute the query, if only one record is updated, then we indicate success by returning true
70         return stmt.executeUpdate() == 1;
```

### 7.1.2 For Query 15 form file:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="UTF-8">
5 <title>Team</title>
6 </head>
7 <body>
8 <h2>Found date</h2>
9 <!--
10 Form for collecting user input for the new movie_night record.
11 Upon form submission, add_movie.jsp file will be invoked.
12 -->
13 <form action=" add_movie1.jsp">
14 <!-- The form organized in an HTML table for better clarity. -->
15 <table border=1>
16 <tr>
17 <th colspan="2">Enter the data:</th>
18 </tr>
19 <tr>
20 <td>Date_T:</td>
21 <td><div style="text-align: center;">
22 <input type=text name=Date_T>
23 </div></td>
24 </tr>
25
26 <tr>
27
28 <tr>
29 <td><div style="text-align: center;">
30 <input type=reset value=Clear>
31 </div></td>
32 <td><div style="text-align: center;">
33 <input type=submit value=Insert>
34 </div></td>
35 </tr>
36 </table>
37 </form>
38 </body>
39 </html>
```

### 7.1.3 For Query 15 Teams file takes input from input form:

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2 pageEncoding="UTF-8"%>
3 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
4 "http://www.w3.org/TR/html4/loose.dtd">
5 <html>
6 <head>
7 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
8 <title>Query Result</title>
9 </head>
10 <body>
11 <%@page import="client_db.DataHandler"%>
12 <%@page import="java.sql.ResultSet"%>
13 <%@page import="java.sql.Array"%>
14 <%
15 // The handler is the one in charge of establishing the connection.
16 DataHandler handler = new DataHandler();
17 // Get the attribute values passed from the input form.
18 String Date_T = request.getParameter("Date_T");
19 /*
20 * If the user hasn't filled out all the time, movie name and duration. This is very
21 simple checking.
22 */
23 if ( Date_T.equals("") ) {
24 response.sendRedirect("add_movie_form1.jsp");
25 } else {
26 int Date_T1 = Integer.parseInt(Date_T);
27
28 // Now perform the query with the data from the form.
29 boolean success = handler.addMovie1(Date_T1);
30 if (!success) { // Something went wrong
31 %>
32 <h2>There was a problem inserting the course</h2>
33 <%
34 } else { // Confirm success to the user
35 %>
36 <h2>Team Table:</h2>
37 <ul>
38
39 <li>Date_T: <%=Date_T%></li>
40
41 </ul>
42 <h2>Was successfully inserted.</h2>
43
44 <a href="get_all_movies.jsp">See Teams Table.</a>
45 <%
46 }
47 }
48 %>
49 </body>
50 </html>
```

### 7.1.4 For Query 1 form file:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="UTF-8">
5 <title>Add Team</title>
6 </head>
7 <body>
8 <h2>Add a new Team</h2>
9 <!--
10 Form for collecting user input for the new movie_night record.
11 Upon form submission, add_movie.jsp file will be invoked.
12 -->
13 <form action=" add_movie.jsp">
14 <!-- The form organized in an HTML table for better clarity. -->
15 <table border=1>
16 <tr>
17 <th colspan="2">Enter the new Team data:</th>
18 </tr>
19 <tr>
20 <td>Team Name:</td>
21 <td><div style="text-align: center;">
22 <input type=text name=T_Name>
23 </div></td>
24 </tr>
25 <tr>
26 <td>Team Type:</td>
27 <td><div style="text-align: center;">
28 <input type=text name=Type>
29 </div></td>
30 </tr>
31 <tr>
32 <td>Date:</td>
33 <td><div style="text-align: center;">
34 <input type=text name=Date_T>
```

```
35 </div></td>
36 </tr>
37 <tr>
38
39 <tr>
40 <td><div style="text-align: center;">
41 <input type=reset value=Clear>
42 </div></td>
43 <td><div style="text-align: center;">
44 <input type=submit value=Insert>
45 </div></td>
46 </tr>
47 </table>
48 </form>
49 </body>
50 </html>
```

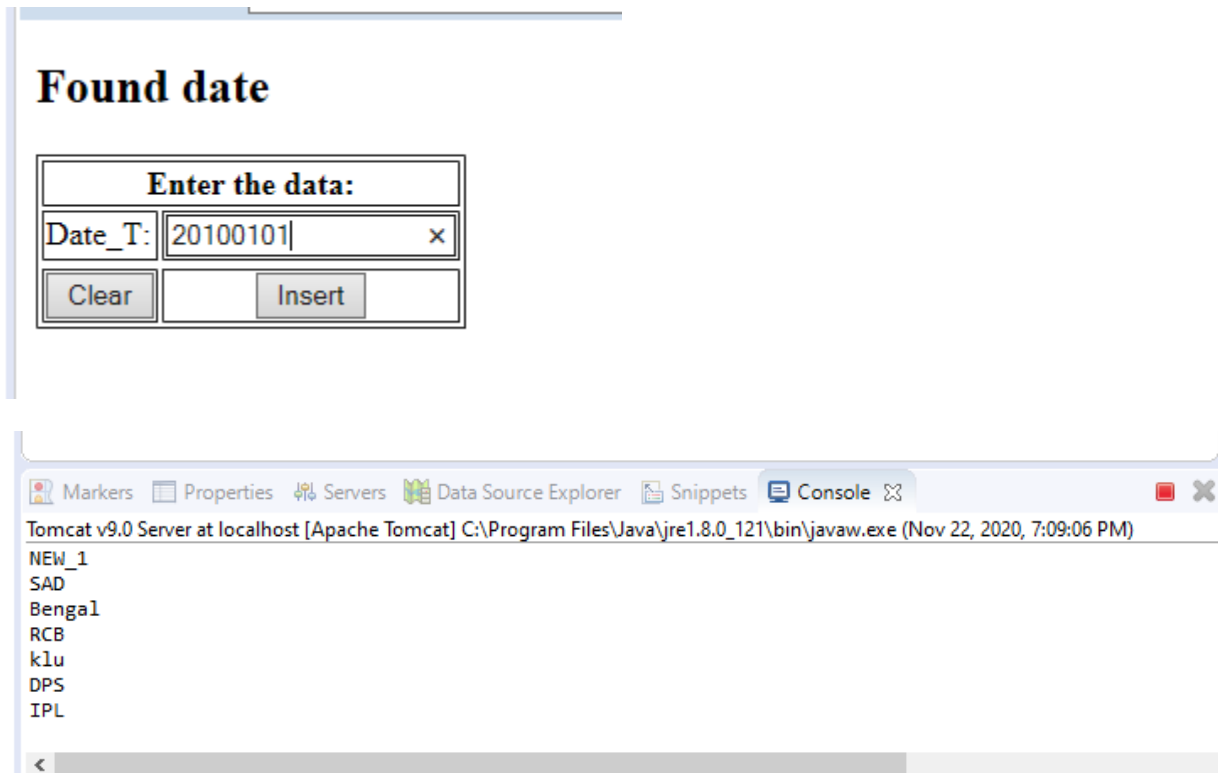


### 7.1.5 For Query 1 Teams file takes input from input form:

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2 pageEncoding="UTF-8"%>
3 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
4 "http://www.w3.org/TR/html4/loose.dtd">
5 <html>
6 <head>
7 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
8 <title>Query Result</title>
9 </head>
10 <body>
11 <%@page import="client_db.DataHandler"%>
12 <%@page import="java.sql.ResultSet"%>
13 <%@page import="java.sql.Array"%>
14 <%
15 // The handler is the one in charge of establishing the connection.
16 DataHandler handler = new DataHandler();
17 // Get the attribute values passed from the input form.
18 String T_Name = request.getParameter("T_Name");
19 String Type = request.getParameter("Type");
20 String Date_T = request.getParameter("Date_T");
21 /*
22 * If the user hasn't filled out all the time, movie name and duration. This is very
23 simple checking.
24 */
25 if (T_Name.equals("") || Type.equals("") || Date_T.equals("")) {
26 response.sendRedirect("add_movie_form.jsp");
27 } else {
28 int Date_T1 = Integer.parseInt(Date_T);
29
30 // Now perform the query with the data from the form.
31 boolean success = handler.addMovie(T_Name, Type, Date_T1);
32 if (!success) { // Something went wrong
33 %>
34 <h2>There was a problem inserting the course</h2>
35 <%
36 } else { // Confirm success to the user
37 %>
38 <h2>Team Table:</h2>
39 <ul>
40 <li>T_Name: <%=T_Name%></li>
41 <li>Type : <%=Type%></li>
42 <li>Date_T: <%=Date_T%></li>
43
44 </ul>
45 <h2>Was successfully inserted.</h2>
46
47 <a href="get_all_movies.jsp">See Teams Table.</a>
48 <%
49 }
50 }
51 %>
52 </body>
53 </html>
```

## 7.2. Screenshots showing the testing of the Web database application

Input from Forms which gets particular date:



7.3) Screenshots to demonstrate Query 1 works in JSP

## Add a new Team

Enter the new Team data:	
Team Name:	<input type="text" value="Last_1"/>
Team Type:	<input type="text" value="Dan"/>
Date:	<input type="text" value="20150101"/> ×
<input type="button" value="Clear"/>	<input type="button" value="Insert"/>

http://localhost:8080/client\_db/add\_movie.

### Team Table:

- T\_Name: Last\_1
- Type : Dan
- Date\_T: 20150101

**Was successfully inserted.**

[See Teams Table.](#)

## 7.4) Screenshots to demonstrate Query 15 works after updating Teams

Input take same input date:

### Found date

<b>Enter the data:</b>	
Date_T:	<input type="text" value="20100101"/> x
<input type="button" value="Clear"/>	<input type="button" value="Insert"/>

The screenshot shows a web browser window with the address bar displaying `http://localhost:8080/client_db/add_movie1.jsp?Date_T=20100101`. The browser tabs include "Query Result", "Team", and "Query Result". The page content displays:

**Team Table:**

- Date\_T: 20100101

**Was successfully inserted.**

[See Teams Table.](#)

Below the browser window, an IDE's console window is visible, showing the output of a Java application. The console text is:

```
Tomcat v9.0 Server at localhost [Apache Tomcat] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (Nov 22, 2020, 7:09:06 PM)
SAD
Bengal
Last_1
RCB
klu
DPS
IPL
```

7.5) Screenshots to demonstrate get\_Teams function works

<b>T_Name</b>	<b>Type</b>	<b>Date_T</b>
Bengal	PC	20150101
DPS	Care	20190403
Her_1	LPS	20140102
IPL	CRIC	20200101
klu	international	20160403
Last_1	Dan	20150101
lpu	college	16670203
NEW_1	JSP	20140105
RCB	POLISH	20150201
SAD	Emotion	20140308