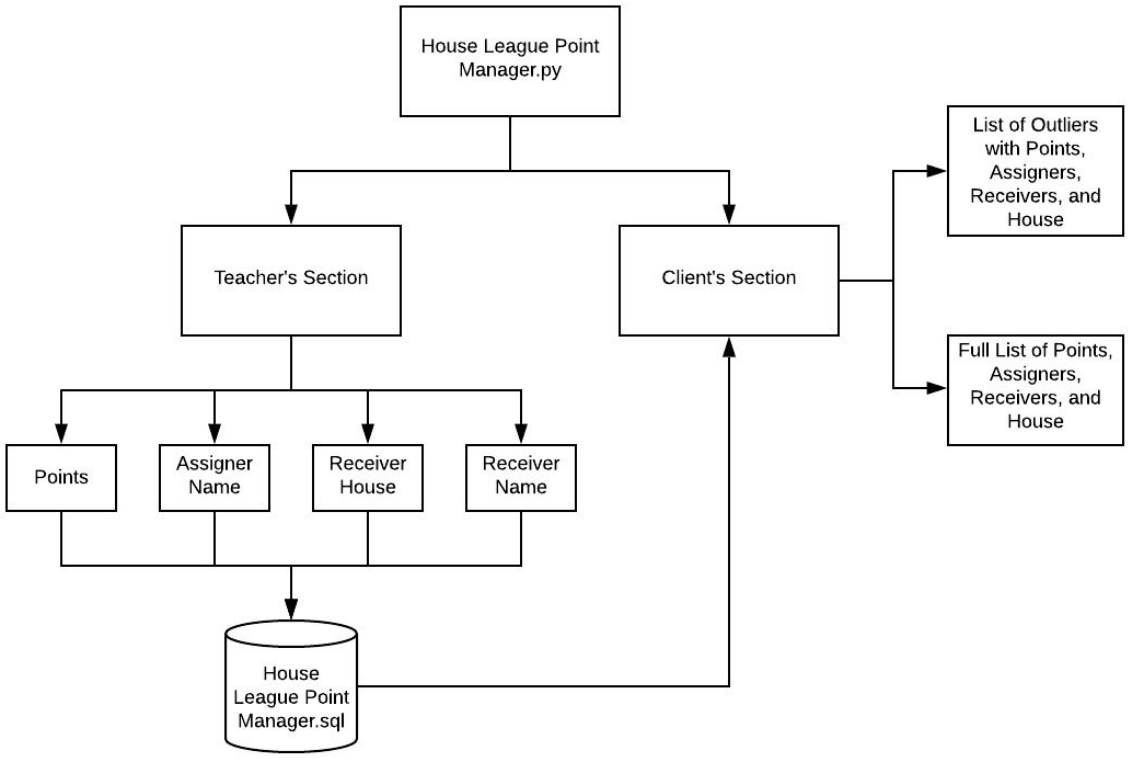
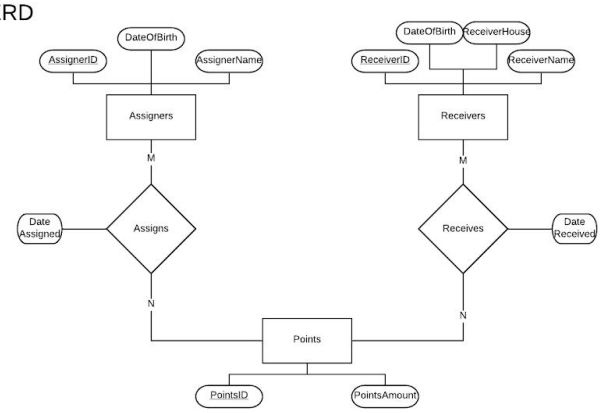
CRITERIA B: Design

System Flowchart

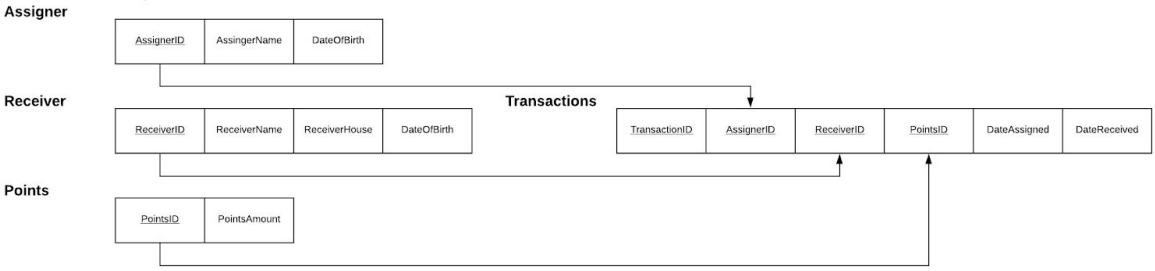


ERD and Relational Map

ERD



Relational Map



Pseudocode

```
DROP TABLE IF EXISTS Transactions;
DROP TABLE IF EXISTS Assigners;
DROP TABLE IF EXISTS Receivers;
DROP TABLE IF EXISTS Points;

CREATE TABLE Assigners
(
	AssignerID INTEGER NOT NULL,
	AssignerName VARCHAR(50) NOT NULL,
	DateOfBirth DATE NOT NULL,
	PRIMARY KEY (AssignerID)
);

CREATE TABLE Receivers
(
	ReceiverID INTEGER NOT NULL,
	ReceiverName VARCHAR(50) NOT NULL,
	ReceiverHouse VARCHAR(10) NOT NULL,
	DateOfBirth DATE NOT NULL,
	PRIMARY KEY (ReceiverID)
);

CREATE TABLE Points
(
	PointsID INTEGER NOT NULL,
	PointsAmount VARCHAR(50) NOT NULL,
	PRIMARY KEY (NovelID)
);

CREATE TABLE Transactions
(
	TransactionID INTEGER NOT NULL,
	AssignerID INTEGER NOT NULL,
	ReceiverID INTEGER NOT NULL,
	PointsID INTEGER NOT NULL,
	DateAssigned DATE NOT NULL,
	DateReceived DATE NOT NULL,
	PRIMARY KEY (TransactionID, AssignerID, ReceiverID, PointsID),
```

```
        FOREIGN KEY (AssignerID) REFERENCES Assigners(AssignerID),
        FOREIGN KEY (ReceiverID) REFERENCES Receivers(ReceiverID),
        FOREIGN KEY (PointsID) REFERENCES Points(PointsID)
);

def getAssigner():
        c.execute("SELECT * FROM Assigners")
        data = c.fetchall()
        return data

def getReceivers():
        c.execute("SELECT * FROM Receivers")
        data = c.fetchall()
        return data

def getPoints():
        c.execute("SELECT * FROM Points")
        data = c.fetchall()
        return data

def getTransactions():
        c.execute("SELECT * FROM Transactions")
        data = c.fetchall()
        return data

def registerAssigner(name, date):
        assigners = getAssigners()
        ins_str = 'INSERT INTO Assigners VALUES (' + str(assigners[-1][0] + 1) + ', "' +
        str(name) + '", "' + str(date) + '");'
        c.execute(ins_str)
        con.commit()

def registerReceiver(name, date):
        receivers = getReceivers()
        ins_str = 'INSERT INTO Receivers VALUES (' + str(receivers[-1][0] + 1) + ', "' +
        str(name) + '", "' + str(date) + '");'
        c.execute(ins_str)
        con.commit()
```

```
def registerPointsAssigned(amount):
        points = getPoints()
        ins_str = 'INSERT INTO Points VALUES (' + str(points[-1][0] + 1) + ', "' +
        str(amount) + '");'
        c.execute(ins_str)
        con.commit()

def logTransaction(assignerID, receiverID, pointsID):
        ins_str = 'INSERT INTO Transactions VALUES (' + str(assignerID) + ', ' +
        str(receiverID) + ', ' + str(pointsID) + ');'
        c.execute(ins_str)
        con.commit()
```

Test Plan

Action to be Tested	Method of Testing
Test if the program runs correctly and the main window appears on the screen	Launch the program
Check if the input box works properly	Type some numbers in the input box
Check if the add button works properly	Add new instance of assigned House points and see if it's in the database
Check if the client can receive the notification	Add an instance into the database with abnormally low/high data, and see if the client can receive the notification
Check if the client is able to check the database	Compare the program database and the original database, add some data to the original database, check the database again to see if it can update

User Interface Design