;

**Project intermediate milestone:** External documentation of data structures, code design, key algorithms, and any additional white box tests.

## Team Members

Meet Nitinbhai Patel

(B00899516)

## Subject:

Software Development Concepts

## Professor:

Mike McAllister

# Data Structure I have used till now and planning to use in the future for the development of project

1. **LinkedList**
    - I have used LinkedList for storing the object of PersonIdentity (which includes personName and personId)
    - I will use LinkedList for storing the object of FileIdentifier (Which includes fileLocation and fileId)
    - I will use it for string nodes and references of an individual
2. **Map**
    - I have used map from storing the person attributes such as “Date of Birth”, “Date of death”, “Birth location”, “Death location”, “Gender” and “Occupation”. I have stored this attribute as strings.
    - I will also use Map for storing media attributes such as “Year”, “date” and “City”. I will store this attribute as string.
3. **Set**
    - Will use set for storing the descendants and ancestors who are within n “generations” generations of the person.
    - Will use it for storing set of media files linked to the given tag and location whose dates fall within the date range
4. **Tree**
    - Will use tree data structure for creating biological family tree.

Will update this file if any data structure will be added or removed in the future milestone.

# Code Design and Algorithm

## Classes

1. **genealogy.java**

**This class give easier access to all the methods.**

**This class will contain**

- **Methods which help to store information about individuals and to store relation between individuals.**
- **Methods required for managing metadata of different forms of media (Video or Picture)**
- **Methods for reporting (answering) several genealogist’s question**

2. **PersonIdentity.java**

**This class is used to identify individuals from the family tree**

- **It contains two variable Int personId and String personName which help to create the object of individual. This object helps use to identify the individual from the family tree.**

3. **FileIdentifier.java**

**This class is used to identify file using fileLocation from media archive (Here we are taking file location instead of file name because name of two file can be same but location of two file is be different )**

- **It contains two variable Int fileId and String fileLocation which help to create the object of each file. This object helps use to identify the file from the media archive.**

4. **biologicalrelation.java**

**This class specifies the degree of removal and degree of cousinship in a relation**

## Database Design

**Following are the tables we need to maintain for family tree management.**

1. **personNameinfo**

| | |
|---|---|
| Int personId (Primary Key) | String personName |

2. **personAttributeInfo**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Int personAttributeId (Primary Key) | String personDOB | String personDOD | String birthLoc | String deathloc | String personGender | String personOccupation | Int fk_personId (foreign Key) |

3. **personReference**

| | |
|---|---|
| int personReferenceId (Primary Key) | String personReference |

4. **personReferenceRelation**

| | |
|---|---|
| int personReferenceId (Foreign Key) | int personId (Foreign Key) |

5. **personNotes**

| | | |
|---|---|---|
| Int personNoteId (Primary Key) | String Notes | Int personId (Foreign Key) |

6. **individualsRelationStatus**

| Int relationId (Primary Key) | String person 1 | String person 2 | String Status |
|---|---|---|---|

**Following are the tables we need to maintain for media archive.**

1. **mediaLocationInfo**

| Int mediaLocationId (Primary Key) | String mediaLocation |
|---|---|

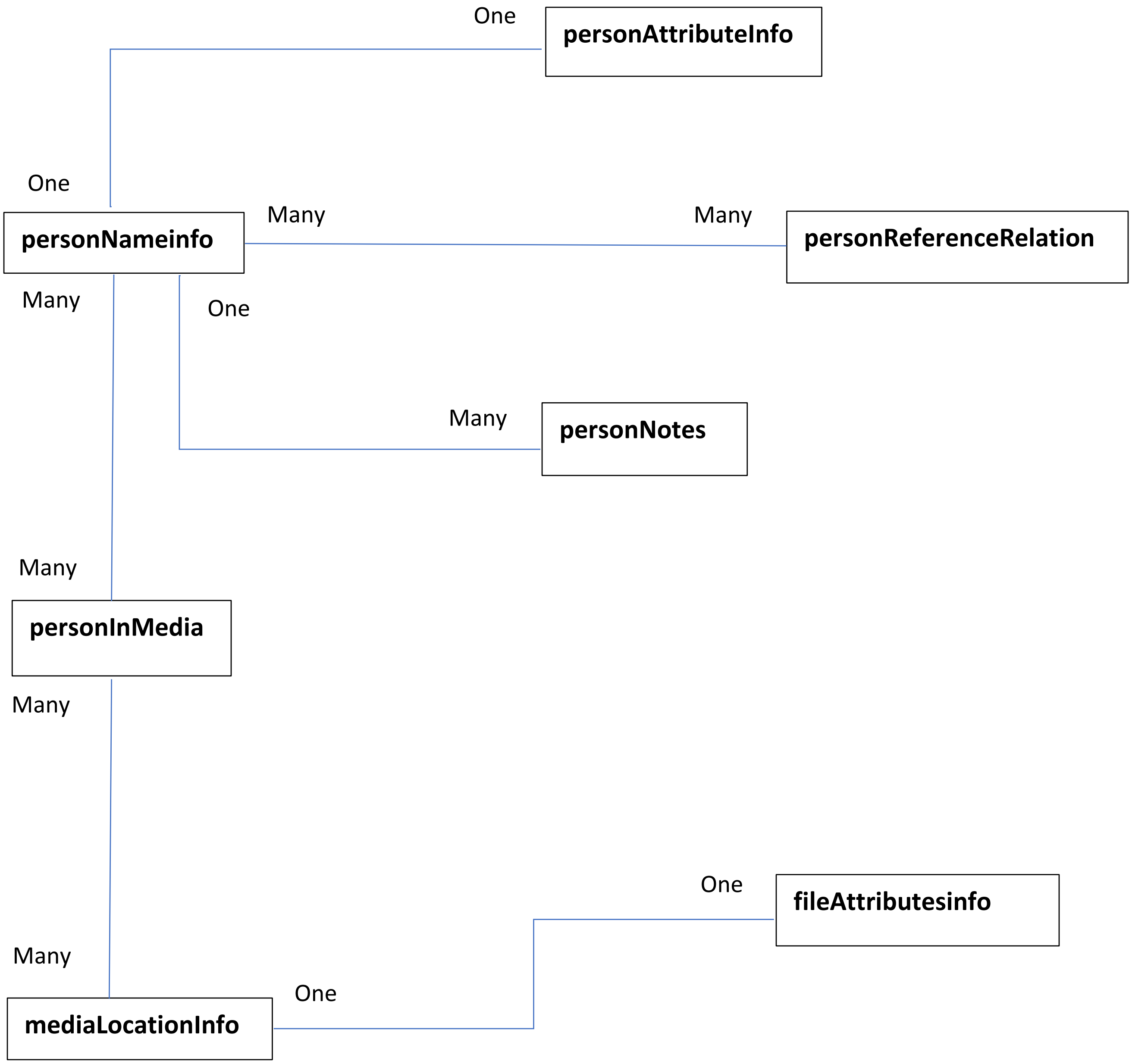
2. **fileAttributesinfo**

| Int mediaAttributeId (Primary Key) | String mediaName | String mediaDate | String location | String mediaTag | int mediaLocationId (Foreign Key) |
|---|---|---|---|---|---|

3. **personInMedia**

| Int mediaLoactionId (Foreign Key) | Int personId (Foreign key) |
|---|---|

## Database Table Relationship



## Test Cases

### Q1) Identify a set of input validation tests for each method as well as the expected outcome for each case. *(Generally, tests on bad input data for which you shouldn't crash)*

### *Family Tree Management*

***Person Name, Birth and Death location, Occupation, Gender, References, Notes***

- String is null
- String is empty
- String contains special character

***Date of Birth and Death***

- String is null
- String is empty

### *Media Archive Management*

***FileName , Picture location, Tags***

- String is null
- String is empty
- String contains special character

***Date when picture taken, File Location on Hard disk***

- String is null
- String is empty

### Q2) Identify a set of boundary tests for these boundary cases. *(Tests at the edge of inputs or problem structures)*

### *Family Tree Management*

***Person Name, Birth and Death location, Occupation, Gender, References, Notes***

- 1 character String
- String Must be within the length if defined

***Date of Birth and Death (Assume that date's datatype is String and is in yyyy-mm-dd formate)***

- Day is 1
- Day is 31
- Month is 1
- Month is 12

## *Media Archive Management*

***FileName , Picture location, Tags, File Location***

- 1 character String
- String Must be within the length if defined

***Date when picture taken***

- Day is 1
- Day is 31
- Month is 1
- Month is 12