Adriance Memorial Library

Faith Mazzone

Poughkeepsie Public Library

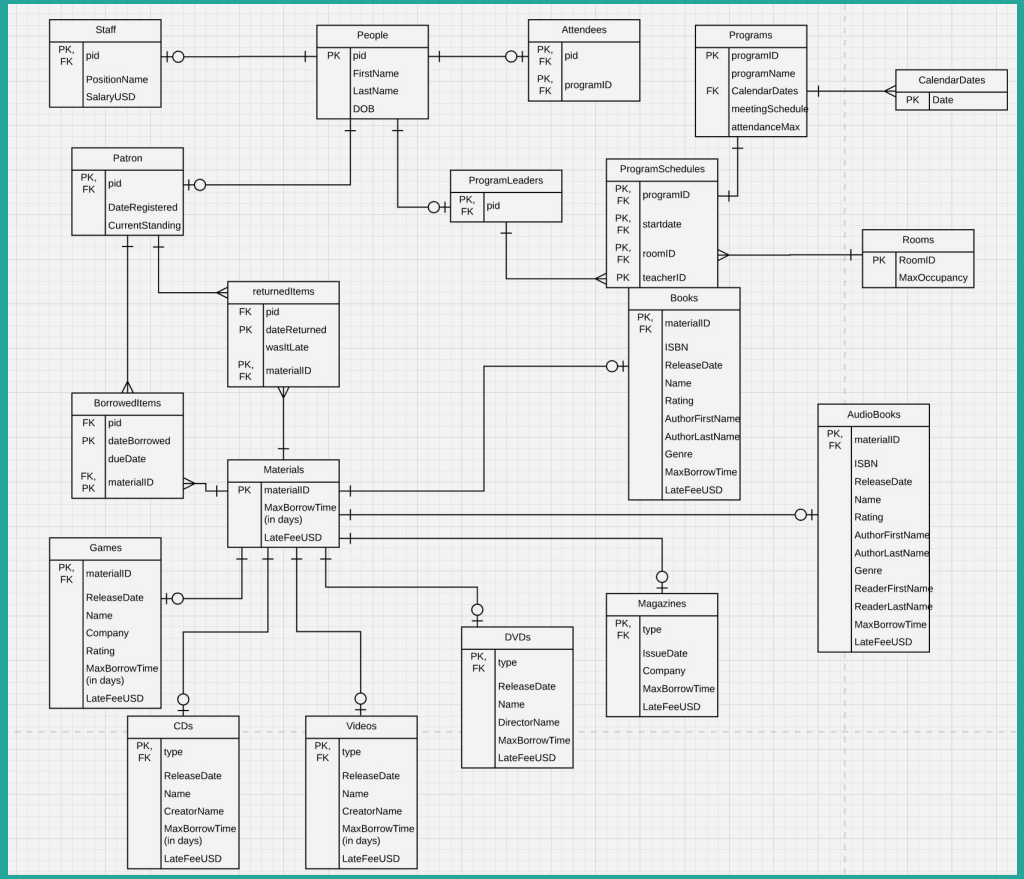
Table of Contents

Executive Summary

This database has been created to manage the many operations that occur at the Adriance Memorial Library in Poughkeepsie.

It keeps track of the various material loans as well as the many clubs and programs that the Library hosts. It marks the separation between patrons, teachers, staff, and attendees, while managing the on-site activities and borrowed items.

ER Diagram


Staff
PK, FK pid
PositionName
SalaryUSD
People
PK pid
FirstName
LastName
DOB
Attendees
PK, FK pid
PK, FK programID
Programs
PK programID
programName
FK CalendarDates
meetingSchedule
attendanceMax
CalendarDates
PK Date
Patron
PK, FK pid
DateRegistered
CurrentStanding
ProgramLeaders
PK, FK pid
ProgramSchedules
PK, FK programID
PK, FK startdate
PK, FK roomID
PK teacherID
Rooms
PK RoomID
MaxOccupancy
returnedItems
FK pid
PK dateReturned
wasItLate
PK, FK materialID
Books
PK, FK materialID
ISBN
ReleaseDate
Name
Rating
AuthorFirstName
AuthorLastName
Genre
MaxBorrowTime
LateFeeUSD
BorrowedItems
FK pid
PK dateBorrowed
dueDate
FK, PK materialID
Materials
PK materialID
MaxBorrowTime (in days)
LateFeeUSD
AudioBooks
PK, FK materialID
ISBN
ReleaseDate
Name
Rating
AuthorFirstName
AuthorLastName
Genre
ReaderFirstName
ReaderLastName
MaxBorrowTime
LateFeeUSD
Games
PK, FK materialID
ReleaseDate
Name
Company
Rating
MaxBorrowTime (in days)
LateFeeUSD
DVDs
PK, FK type
ReleaseDate
Name
DirectorName
MaxBorrowTime
LateFeeUSD
Magazines
PK, FK type
IssueDate
Company
MaxBorrowTime
LateFeeUSD
CDs
PK, FK type
ReleaseDate
Name
CreatorName
MaxBorrowTime (in days)
LateFeeUSD
Videos
PK, FK type
ReleaseDate
Name
CreatorName
MaxBorrowTime (in days)
LateFeeUSD

People Table

The people table contains all general information about a person in the database, regardless of their relations.

```
CREATE TABLE People (
        PID char(4) not null,
        FirstName text,
        LastName text,
        DOB date,
        primary key (PID)
);
```

	pid character(4)	firstname text	lastname text	dob date
1	p001	Booker	DeWitt	1874-04-19
2	p002	Nicole	Greyson	1985-03-02
3	p003	Faith	Mazzone	1997-06-08
4	p007	Alan	Labouseur	1992-01-01
5	p004	Katie	Bradford	2002-12-05
6	p006	Tyler	Wheatley	2007-03-31
7	p008	Olivia	Bleasdale	1982-07-08
8	p009	Jack	Nguyen	1996-11-13
9	p010	Jeffrey	Parker	1999-05-23
10	p011	Joey	Hanson	1981-09-17

Functional Dependencies:

PID -> FirstName, LastName, DOB

Patrons Table

This table describes the actual Library Card holders, known as members or patrons. It marks when they registered as a member and their standing with the library. A patron can be loaned materials.

```
CREATE TABLE Patrons (
        PID char(4) not null references People(PID),
        DateRegistered date,
        CurrentStanding text CHECK (CurrentStanding = 'Good'
                                    or CurrentStanding = 'Bad'),
        primary key (PID)
);
```

	pid character(4)	dateregistered date	currentstanding text
1	p002	2014-05-01	Good
2	p006	2016-06-21	Bad
3	p007	2015-11-10	Good
4	p004	2017-03-01	Good
5	p008	2012-08-14	Good
6	p009	2017-01-05	Good
7	p010	2013-09-14	Good
8	p011	2017-04-15	Good

Functional Dependencies:

PID -> dateRegistered, currentStanding

Staff Table

This table contains data only intended for the staff members that would not be applicable to other denominations. Staff must also be a Patron if they wish to be loaned materials.

```
CREATE TABLE Staff (
        PID char(4) not null references People(PID),
        positionName text,
        SalaryUSD int,
        primary key (PID)
);
```

	pid character(4)	positionname text	salaryusd integer
1	p002	Librarian	59350
2	p011	Maintenance	30780

Functional Dependencies:

PID -> positionName, SalaryUSD

Attendees Table

This table documents club and program attendees. One person can attend many clubs.

```
CREATE TABLE Attendees (
        PID char(4) not null references People(PID),
        ProgramID char(4) not null references Programs(ProgramID),
        primary key (PID, ProgramID)
);
```

	pid character(4)	programid character(4)
1	p004	g001
2	p006	g001
3	p008	g001
4	p009	g001
5	p011	g002
6	p004	g002
7	p009	g003
8	p010	g003
9	p004	g003

PID, ProgramID ->

ProgramLeaders Table

This table describes the teachers and leaders of the programs at the Library. To be registered as a Leader, one must undergo a background check. A ProgramLeader cannot be loaned materials unless they are also registered as Patrons.

```
CREATE TABLE ProgramLeaders (
        PID char(4) not null references People(PID),
        primary key (PID)
);
```

	pid character(4)
1	p003
2	p011
3	p008

Functional Dependencies:

PID ->

CalenderDates Table

This table is used to describe weekly timeslots. It would not be used for one-time events.

```
CREATE TABLE CalendarDates (
        calID char(8),
        dayName text CHECK (dayName = 'Monday'
                        or dayName = 'Tuesday'
                        or dayName = 'Wednesday'
                        or dayName = 'Thursday'
                        or dayName = 'Friday'
                        or dayName = 'Saturday'
                        or dayName = 'Sunday'),
        startTimeHour int,
        startTimeMin int,
        primary key (calID)
);
```

	calid character(8)	dayname text	starttimehour integer	starttimemin integer
1	cal00001	Monday	15	30
2	cal00002	Saturday	10	0
3	cal00003	Wednesday	18	15
4	cal00004	Friday	12	30

Functional Dependencies:

calID -> dayName, startTimeHour, startTimeMin

Rooms Table

This table describes all of the rooms available at the library for events or programs.

```
CREATE TABLE Rooms (
        RoomID char(4),
        RoomName text,
        MaxOccupancy int,
        primary key (RoomID)
);
```

	roomid character(4)	roomname text	maxoccupancy integer
1	r001	Teen Study Room	30
2	r002	Charwat Meeting Room	50
3	r003	Small Periodicals Room	15
4	r004	Greenspan Board Room	23
5	r005	Ground Floor Lobby	150

Functional Dependencies:

roomID -> roomName, MaxOccupancy

Programs Table

This table documents the programs themselves, as well as their weekly timeslots.

```
CREATE TABLE Programs (
        ProgramID char(4),
        ProgramName text,
        CalendarDates char(8) references CalendarDates(calID),
        primary key (ProgramID)
);
```

	programid character(4)	programname text	calendardates character(8)
1	g001	Girls Who Code	cal00002
2	g002	Gay, Lesbian, and Straight Education Network	cal00004
3	g003	Knitting and Crocheting Club	cal00001

Functional Dependencies:

programID -> programName, calendarDates

ProgramSchedules Table

This table keeps track of the programs actually taking place, as well as their locations and leaders.

```
CREATE TABLE ProgramSchedules (
        ProgramID char(4) not null references Programs(ProgramID),
        StartDate date,
        RoomID char(4) references Rooms(RoomID),
        teacherPID char(4) references ProgramLeaders(PID),
        primary key (programID, startDate)
);
```

	programid character(4)	startdate date	roomid character(4)	teacherpid character(4)
1	g001	2017-09-11	r001	p003
2	g002	2016-11-01	r002	p008
3	g003	2017-02-20	r004	p011

Functional Dependencies:

programID, startDate -> roomID, teacherID

Materials Table

This table documents the general information about the materials available for loan at the Library.

```
CREATE TABLE Materials (
        materialID char(8) not null,
        primary key (materialID)
);
```

	materialid character(8)
1	m0000001
2	m0000002
3	m0000003
4	m0000004
5	m0000005
6	m0000006
7	m0000007
8	m0000008
9	m0000009

Functional Dependencies:

materialID ->

Books Table

This table describes the Book materials available at the library.

```
CREATE TABLE Books (
        materialID char(8) not null references Materials(materialID),
        MaxBorrowTimeInDays int,
        LateFeePerDayUSD float,
        ISBN varchar(16),
        ReleaseDate date,
        Name text,
        AuthorFirstName text,
        AuthorLastName text,
        Genre text,
        primary key (materialID)
);
```

	materialid character(8)	maxborrowtimeindays integer	latefeeperdayusd double precision	isbn character varying(16)	releasedate date	name text	authorfirstname text	authorlastname text	genre text
1	m0000001	21	0.1	159514188X	2011-06-14	13 Reasons Why	Jay	Asher	YA Fiction
2	m0000007	21	0.1	9788496581579	1985-01-15	Enders Game	Orson	Card	SciFi

Functional Dependencies:

materialID -> MaxBorrowTimeInDays, LateFeePerDayUSD, ISBN, ReleaseDate, Name, AuthorFirstName, AuthorLastName, Genre

AudioBooks Table

This table describes the AudioBook materials available at the library.

```
CREATE TABLE AudioBooks (
        materialID char(8) not null references Materials(materialID),
        MaxBorrowTimeInDays int,
        LateFeePerDayUSD float,
        ISBN varchar(16),
        ReleaseDate date,
        Name text,
        AuthorFirstName text,
        AuthorLastName text,
        Genre text,
        ReaderFirstName text,
        ReaderLastName text,
        primary key (materialID)
);
```

	materialid character(8)	maxborrowtimeindays integer	latefeeperdayusd double precision	isbn character varying(16)	releasedate date	name text	authorfirstname text	authorlastname text	genre text	readerfirstname text	readerlastname text
1	m0000002	21	0.1	9780525618331	2017-11-01	The Midnight Line	Lee	Child	Mystery	Dick	Hill

Functional Dependencies:

materialID -> MaxBorrowTimeInDays, LateFeePerDayUSD, ISBN, ReleaseDate, Name, AuthorFirstName, AuthorLastName, Genre, ReaderFirstName, ReaderLastName

Magazines Table

This table describes the Magazine materials available at the library.

```
CREATE TABLE Magazines (
        materialID char(8) not null references Materials(materialID),
        MaxBorrowTimeInDays int,
        LateFeePerDayUSD float,
        issueDate date,
        Publisher text,
        primary key (materialID)
);
```

	materialid character(8)	maxborrowtimeindays integer	latefeeperdayusd double precision	issuedate date	publisher text
1	m0000003	21	0.1	2016-03-01	Marie Claire
2	m0000008	21	0.1	2015-02-23	TIME

Functional Dependencies:

materialID -> MaxBorrowTimeInDays, LateFeePerDayUSD, issueDate, Publisher

CDs Table

This table describes the CD materials available at the library.

```
CREATE TABLE CDs (
        materialID char(8) not null references
Materials(materialID),
        MaxBorrowTimeInDays int,
        LateFeePerDayUSD float,
        ReleaseDate date,
        Name text,
        CreatorName text,
        primary key (materialID)
);
```

	materialid character(8)	maxborrowtimeindays integer	latefeeperdayusd double precision	releasedate date	name text	creatorname text
1	m0000004	7	0.25	1997-05-26	N Sync	NSYNC

Functional Dependencies:

materialID -> MaxBorrowTimeInDays, LateFeePerDayUS, ReleaseDate, Name, CreatorName

DVDs Table

This table describes the DVD materials available at the library.

```
CREATE TABLE DVDs (
        materialID char(8) not null references Materials(materialID),
        MaxBorrowTimeInDays int,
        LateFeePerDayUSD float,
        ReleaseDate date,
        Name text,
        Rating text,
        DirectorFirstName text,
        DirectorLastName text,
        primary key (materialID)
);
```

	materialid character(8)	maxborrowtimeindays integer	latefeeperdayusd double precision	releasedate date	name text	rating text	directorfirstname text	directorlastname text
1	m0000005	7	1	2005-03-15	The Incredibles	PG	Brad	Bird

Functional Dependencies:

materialID -> MaxBorrowTimeInDays, LateFeePerDayUSD, ReleaseDate, Name, Rating, DirectorFirstName, DirectorLastName

Games Table

This table describes the Games materials available at the library.

```
CREATE TABLE Games (
        materialID char(8) not null references Materials(materialID),
        MaxBorrowTimeInDays int,
        LateFeePerDayUSD float,
        ReleaseDate date,
        Name text,
        MinAge int,
        Publishers text,
        primary key (materialID)
);
```

	materialid character(8)	maxborrowtimeindays integer	latefeeperdayusd double precision	releasedate date	name text	minage integer	publishers text
1	m0000006	7	0.25	1991-01-01	Sorry!	6	Hasbro
2	m0000009	7	0.25	2003-01-01	Garfields Typing Pal	7	Typing Pal

Functional Dependencies:

materialID -> MaxBorrowTimeInDays, LateFeePerDayUSD, ReleaseDate, Name, MinAge, Publishers

BorrowedItems Table

This table documents all of the items borrowed. It relies on the material and the date it was borrowed.

```
CREATE TABLE BorrowedItems (
        PID char(4) not null references Patrons(PID),
        dateBorrowed date,
        dueDate date,
        materialID char(8) references Materials(materialID),
        primary key (dateBorrowed, materialID)
);
```

	pid character(4)	dateborrowed date	duedate date	materialid character(8)
1	p002	2017-12-11	2017-12-25	m0000001
2	p006	2017-02-10	2017-02-17	m0000009
3	p006	2017-03-21	2017-03-28	m0000005
4	p011	2016-08-02	2016-08-23	m0000001
5	p007	2017-01-17	2017-02-03	m0000007
6	p010	2017-04-05	2017-04-12	m0000006
7	p010	2014-01-13	2014-12-25	m0000007
8	p004	2017-03-08	2017-03-15	m0000006
9	p010	2013-10-23	2013-10-30	m0000004
10	p007	2015-11-19	2015-12-10	m0000008
11	p006	2017-01-02	2017-01-09	m0000009

Functional Dependencies:

dateBorrowed, materialID -> PID, dueDate

ReturnedItems Table

This table documents the items that have been successfully returned, and whether or not they were late.

```
CREATE TABLE ReturnedItems (
        PID char(4) not null references Patrons(PID),
        dateReturned date,
        wasItLate boolean,
        materialID char(8) references Materials(materialID),
        primary key (dateReturned, materialID)
);
```

	pid character(4)	datereturned date	wasitlate boolean	materialid character(8)
1	p002	2017-12-13	f	m0000001
2	p006	2017-03-10	t	m0000009
3	p011	2016-08-20	f	m0000001
4	p007	2017-02-03	f	m0000007
5	p010	2017-04-10	f	m0000006
6	p010	2014-12-17	f	m0000007
7	p004	2017-03-13	f	m0000006
8	p010	2013-12-10	f	m0000004
9	p007	2015-12-08	f	m0000008
10	p006	2017-01-09	f	m0000009

dateReturned, materialID -> PID, wasItLate

Views

Entirely unused materials:

```
create or replace view unusedMaterials
        as select distinct m.materialID
        from Materials m, BorrowedItems b
        where m.materialId not in
                (select materialID from BorrowedItems)
        order by m.materialID ASC;
```

```
select * from unusedMaterials;
```

	materialid character(8)
1	m0000002
2	m0000003

Stored Procedure

Get Patron's Checkout History

```
CREATE OR REPLACE FUNCTION getPatronsCheckoutHistory (text, text, REFCURSOR) RETURNS refcursor AS
$$
        DECLARE
                newFirstName TEXT := $1;
                newLastName TEXT := $2;
                resultset REFCURSOR := $3;
BEGIN
        OPEN resultset FOR
                select distinct b.materialID, b.dateBorrowed
                from BorrowedItems b, Patrons p
                where b.PID =
                        (select distinct PID
                        from People p
                        where p.FirstName LIKE newFirstName and p.LastName LIKE newLastName);
        return resultset;
END
$$
LANGUAGE plpgsql;
```

```
SELECT getPatronsCheckoutHistory('Jeffrey',
'Parker', 'ref');
FETCH ALL FROM ref;
```

	materialid character(8)	dateborrowed date
1	m0000004	2013-10-23
2	m0000006	2017-04-05
3	m0000007	2014-01-13

```
SELECT getPatronsCheckoutHistory('A%', 'L%',
'ref');
FETCH ALL FROM ref;
```

	materialid character(8)	dateborrowed date
1	m0000007	2017-01-17
2	m0000008	2015-11-19

Reports

Number of new members in 2017

```
select count(pid)
from Patrons
where Patrons.dateRegistered >= '2017-01-01';
```

Number of loans taken out in 2017

```
select count(materialID)
from BorrowedItems
where dateBorrowed >= '2017-01-01';
```

User Roles

```
create role admin;
create role management;
create role librarian;
create role frontDesk;
```

```
grant all on all tables in schema public to admin;
```

```
grant SELECT, INSERT, DELETE on People to management;
grant SELECT, INSERT, DELETE on Patrons to management;
grant SELECT, INSERT, DELETE on ProgramLeaders to management;
grant SELECT, INSERT, DELETE on CalendarDates to management;
grant SELECT, INSERT, DELETE on Programs to management;
grant SELECT, INSERT, DELETE on Attendees to management;
grant SELECT, INSERT, DELETE on ProgramSchedule to management;
grant SELECT, INSERT on Materials to management;
grant SELECT, INSERT on Books to management;
grant SELECT, INSERT on AudioBooks to management;
grant SELECT, INSERT on Magazines to management;
grant SELECT, INSERT on CDs to management;
grant SELECT, INSERT on DVDs to management;
grant SELECT, INSERT on Games to management;
grant SELECT on BorrowedItems to management;
grant SELECT on ReturnedItems to management;
```

```
grant SELECT, INSERT on People to librarian;
grant SELECT, INSERT on Patrons to librarian;
grant SELECT, INSERT on ProgramLeaders to librarian;
grant SELECT, INSERT on CalendarDates to librarian;
grant SELECT, INSERT on Programs to librarian;
grant SELECT, INSERT on Attendees to librarian;
grant SELECT, INSERT on ProgramSchedule to librarian;
grant SELECT, INSERT on Materials to librarian;
grant SELECT, INSERT on Books to librarian;
grant SELECT, INSERT on AudioBooks to librarian;
grant SELECT, INSERT on Magazines to librarian;
grant SELECT, INSERT on CDs to librarian;
grant SELECT, INSERT on DVDs to librarian;
grant SELECT, INSERT on Games to librarian;
grant SELECT, INSERT on BorrowedItems to librarian;
grant SELECT, INSERT on ReturnedItems to librarian;
```

```
grant SELECT on People to frontDesk;
grant SELECT on Patrons to frontDesk;
grant SELECT on ProgramLeaders to frontDesk;
grant SELECT on CalendarDates to frontDesk;
grant SELECT on Programs to frontDesk;
grant SELECT on Attendees to frontDesk;
grant SELECT on ProgramSchedule to frontDesk;
grant SELECT on Materials to frontDesk;
grant SELECT on Books to frontDesk;
grant SELECT on AudioBooks to frontDesk;
grant SELECT on Magazines to frontDesk;
grant SELECT on CDs to frontDesk;
grant SELECT on DVDs to frontDesk;
grant SELECT on Games to frontDesk;
grant SELECT, INSERT on BorrowedItems to frontDesk;
grant SELECT, INSERT on ReturnedItems to frontDesk;
```

Known Problems/Enhancements

The materials need to be stored better to more easily search through them.

Certain restrictions have not been applied on this system - for example, for a Patron's first six months, they cannot take out more than 5 materials at a time. This has not been implemented yet.

I am not satisfied with how the BorrowedItems and ReturnedItems tables interact, I feel it should be more seamless and would work to make it easier to see which items are still 'out'.

A more automatic system should be in place to register late returns.