University of Puerto Rico - Río Piedras
Department of Computer Science

Prison Fellowship of Puerto Rico

Bianca I. Colón Rosado
CCOM 4027 002 - 2016
https://github.com/BnkColon/ccpr-database.git

The Prison Fellowship of Puerto Rico is a nonprofit organization that bring lasting transformation to men, women, and children impacted by incarceration in Puerto Rico. The organization is composed by volunteers, people that donate their time to do services without payment. This organization is growing exponentially every year. The problem is that they save the information about their volunteers in Excel tables. Initially was useful because they normally have as much twenty volunteers in the organization and was easy find information in Excel. But in the past two years they grown from twenty to three hundred fifty volunteers and continues growing over the months. Now is getting complicated for them to find or share information between the administrative people and their own volunteers in case to be necessary.

For that reason we are providing this database. In this database they can save the volunteers information and other information about the organization like the prisons they are working, board members or inclusive the prisoners and their child's, information they normally have in different Excel tables that is hard to merge and analyse.

Using this database the organization would be able to find, iterate and share all the information or specific information about specific cases. Also they would be able to provide statistics about the organization for example how many volunteers, how many womens or men. We notice that they create a map of the Island of Puerto Rico divided in six zones, and they use this map to locate the volunteers per zone. We implate this zones in our database as an attribute of the volunteer information, and now they can do statistics about the volunteers they have per zone. Also they can do estimates about the money could receive per year from their volunteers/chaplains, and many other searches that was difficult to made using Excel or without expone other delicate information like phone numbers, address, child names, or any other information they don't want to share with everybody. We accomplish all the user requirements.

Analysis:

The Prison Fellowship has a group of volunteers to realize activities that helps with the rehabilitation of prisoners, ex-prisoners and victims. Some of our volunteers visit the different prisons of the island to provide courses called discipleship, these courses have a spiritual formation and enters the moral issues, depression, forgiveness, guilt, feelings, values and concerns among others. Other of our programs is Angel Tree, the purpose of this program is send a present in Christmas to the son or daughter of the prisoner from their parent (prisoner) that was in the prison. We save the amount of children's, gender and ages of the cases we work as statistics for the organization.

To help the volunteers we try to accommodate the volunteer to work in a prison near to their house, because they don't get paid for they labor hours. For that reason is important save the volunteer living zone, and we can get that information in a map provided by the organization. After the volunteer is assigned to one prison we need to know the hours and days the of that visit. A volunteer can visit one, many or any prisons. Also the volunteer can work different cases with prisoners and they childs.

We also have a chaplain course, is important to difference the volunteers of the chaplains. With that differentiation we can assign different cases to the chaplains and volunteers. The chaplains annually pay more than a volunteer because they pay the renovation of the chaplain id. By counting the chaplains and volunteers we can estimate the monetary income for next year, and with that estimate we can plan our activities.

User Requirements:

- Need a list of volunteers, chaplains, members of the board of directors.
- Have a list of all the prisons of the island.
- Identify the cases of the prisoners they work
- Get quantity, ages, gender of children cases worked for the Angel Tree program.
- Need record of the days and hours the volunteer visit prison.
- Know the occupation of our volunteers to be effective with the assigns of work and where they live.
- Count volunteers and chaplains in the organization.

Conceptual Design:

After read the analysis and the user requirements we can make a preliminary conceptual design that can be modified during this project. This is because in the process we can find missing elements or errors when designing specific transactions.

We can identify some entities and start looking for the attributes for those entities. As first look we identify the entity of Volunteer, Prisons, Chaplains and Board of Directors. For the Volunteer entity we will need name, last name, address, gender, church, occupation and interest. For the chaplain entity we have the volunteer ID and a chaplain bool to know if is or not a chaplain. Also we need information about the prisons, we need prison name, address and gender. For the board of members we need to identify the position, volunteer ID and the year.

The purpose of the design phase is to start the implementation process of the needs provided by the Prison Fellowship of Puerto Rico. In this phase we identify the relations between entities the particular information of each one. How they connect with each other and what information will be redundant or unnecessary. Once we start working we can assert our initial hypothesis or make some modifications that would help the effectiveness of our database.

For example, we start thinking that we need a Chaplain entity. But when we start working we notice that a every chaplain have to be a volunteer, and the chaplain don't have any particular attribute. For that reason we include an bool attribute to the entity Volunteer called chaplain.

Business rules:

- Volunteer visit one, many or none prisons.
- The volunteer can live near the prison where they visit, or can not.
- Every volunteer can have one, many or none prisoners.
- Every prisoner has one or more volunteers.
- The prisoner have to be assigned to one prison.
- Every child has a prisoner.
- Every Board has one or more volunteers.
- Volunteer can be part of many Board or can not.
- Record the day and time the volunteer visit a prison.
- The Board have one period of time assigned.
- Every volunteer can have only one position per period
- One position can be assigned to many volunteers.
- Each volunteer and prisoner must include the living zone

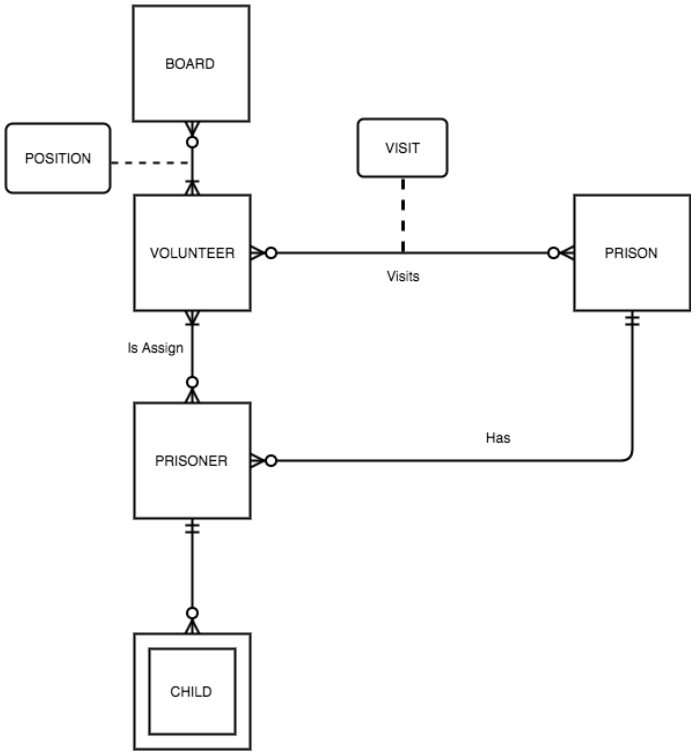
Entities and its attributes:

- Entities:
  - *Volunteer* will save the information about the person that is part of the organization.
  - *Prison* will have information about all the jails in the island of Puerto Rico.
  - *Prisoner* contain the information about the prisoners that are attended by our volunteers.
  - *Child* information is about the childs of the prisoners. We will save basic information about the child cases we work in the Angel Tree Program for statistics.
  - *Period* will contain the period of time of the board. At least the start date.
  - *Position* save the information about the positions of the board, including title, description and terms.

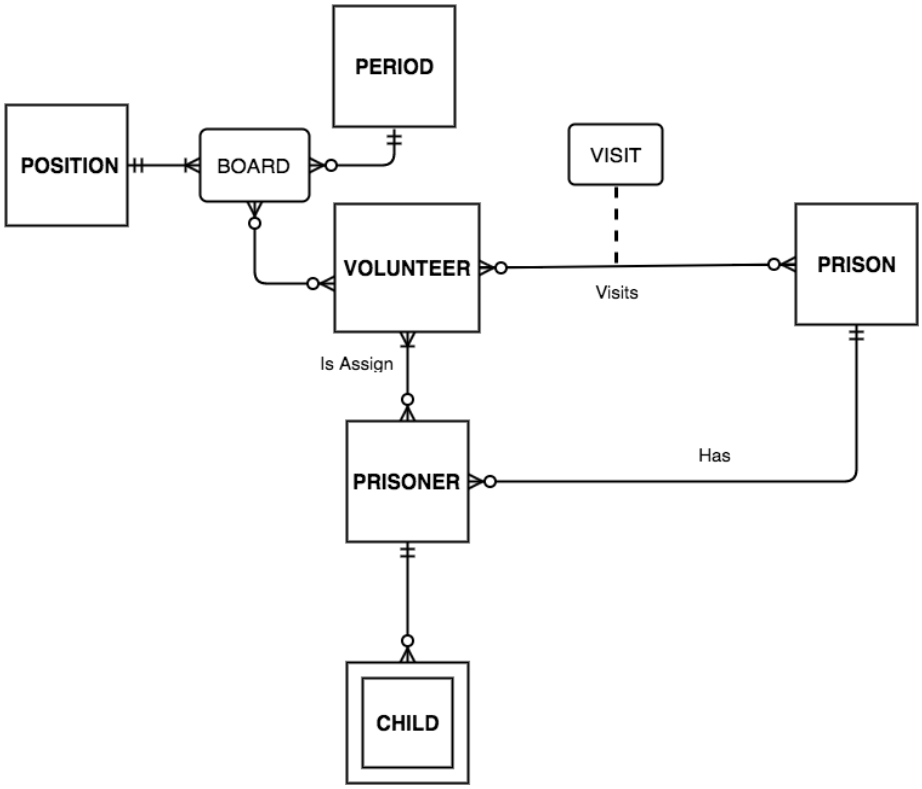
- Attributes:
  - Volunteer:
    - volunteerID - With this number we can identify every volunteer.
    - volunteerName - The name of the volunteer.
    - volunteerLastName - The last name of the volunteer.
    - volunteerAddress - This address will be use to locate their job zone.
    - volunteerCity - This city will be use to locate their job zone, and for statistics.
    - volunteerZipCode - This zip code will be use to locate their job zone.
    - volunteerChurch - The organization need the pastoral endorse. Also is useful to assign prisons.
    - volunteerGender - Is required for records.
    - volunteerPhone - Contact information.
    - volunteerOccupation - Useful to assign cases.
    - volunteerChaplain - Bool, for statistics.
    - volunteerInterest - Useful to assign cases.
    - volunteerZone - The organization divide the island in six zones. They can view where is the percent of volunteers per zone.
  - Period:
    - periodID - Identify every period
    - periodInitial - For records, to identify the start date of every period.
    - periodLast - Is optional, because we can estimate the end date but can change.
  - Prison:
    - prisonID - Is required to identify the prison.
    - prisonName - Is required, to identification.
    - prisonAddress - To locate volunteers near the prison.
    - prisonCity - To locate volunteers near the prison.
    - prisonZipCode - To locate volunteers near the prison.
    - prisonGender - Gender of the prison is required to assign volunteers of the same gender.

- Prisoner:
    - prisonerID - Is required to identify the prisoner.
    - prisonerName - For identification.
    - prisonID - The prisoner have to be in a prison.
    - volunteerID - They volunteer who works in the case.
    - prisonerGender - Is required to assign volunteers of the same gender
    - prisonerZone - Useful to view where is the percent of prisoners per zone.
- Child:
    - prisonerID - Identify the parent of this child.
    - childName - Identify the child.
    - childAge - Is required for statistics.
    - childGender - Is required for statistics.
    - childAddress - Help locate a volunteer near the area.
- Position:
    - positionID - Identify every position.
    - positionName - Job title.
    - positionDescription - Job description.
    - positionTerms - With this number we know when the position expired.

Entity-relationship diagram (E-R diagram):



This is the first E-R diagram we worked. It change because when we start working in the logical design, we notice that the attributes obtained from the relation between VOLUNTEER and BOARD, that we call POSITION, containing the *position* in the board, *description* of the position and *terms* of the position, will be redundant in the way was implemented. For that reason we create a new entity called POSITION that will include this attributes: positionID, positionName, positionDescription and positionTerms.

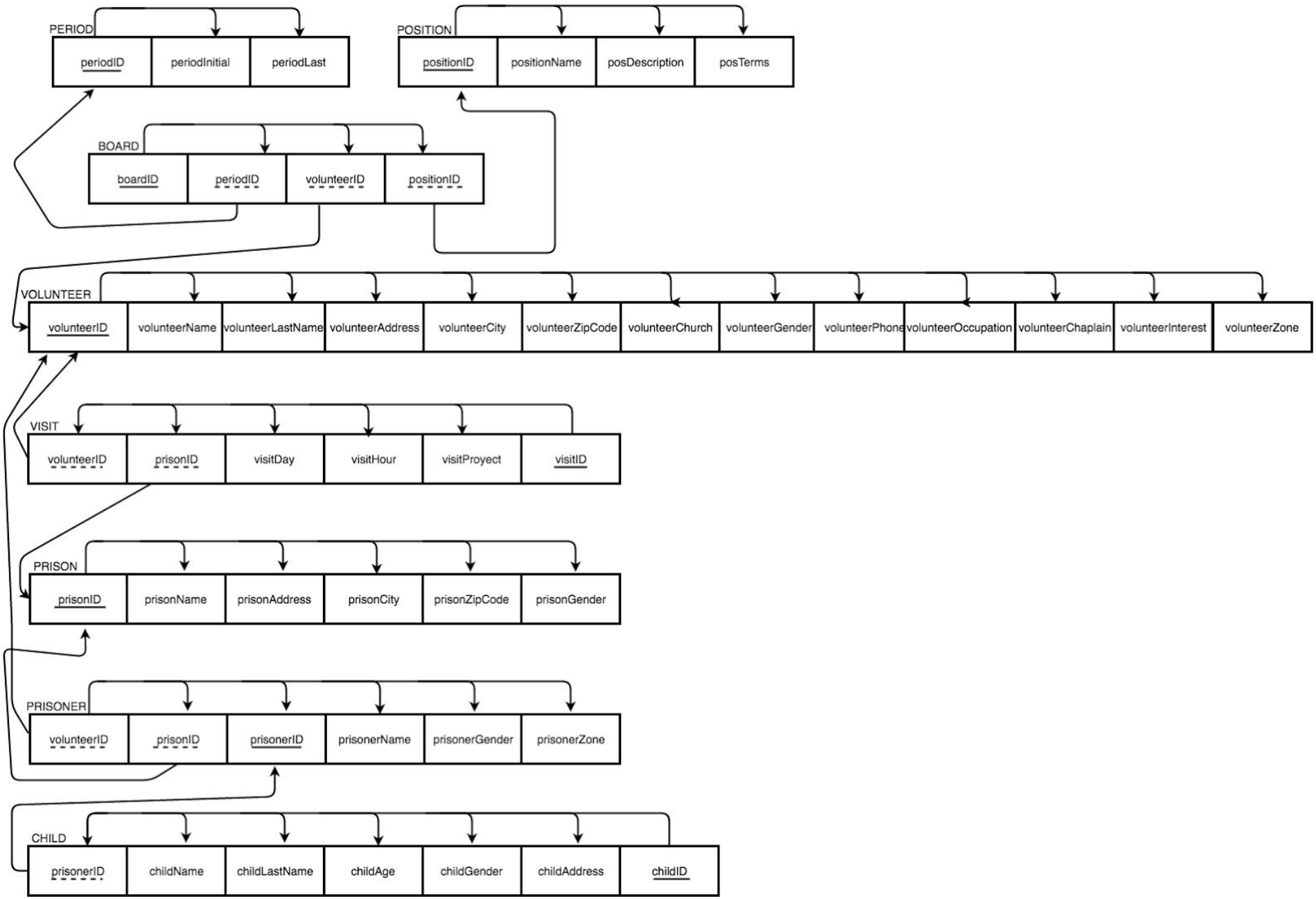


Logical Design:

The purpose of the logical design is to transform the conceptual schema into a logical schema, which describes the data in terms of the data management technology that will be used to implement our database. The conceptual data model is transformed and represented using elements of the relational model which include tables, columns, rows, primary keys, foreign keys, and constraints. The final step in logical database design is to transform the combined and reconciled data specifications into basic, or atomic, elements following well established rules for well-structured data specifications.

When we start working in this phase, we notice that some of our attributes generate redundant information in the database. That's the case we explain when we change the E-R diagram. Also we include ID's in entities that initially don't have, for example the Child weak entity, we prognosticate that the combination of prisoner id, child's name and child's last name will be enough to identify a child assuming that a parent will not put the same name to their child's, but eventually we create a child id. The action to create an id in a weak entity was obtained from the book, Modern Database Management tenth edition on page 168, in a similar example, they named this action a surrogate primary key.

After transforming the E-R diagram to a Logical Design and add identifiers to all the entities we find that the associative entity that we called BOARD, was converted to a table that has Position id, Period id and Volunteer id as foreign keys. Also the relation attribute that we call VISIT was converted to a table with volunteer id and prison id as they foreign keys. The table VOLUNTEER don't have any foreign keys, and the Volunteer id is the primary key. The PRISON table have a primary key in the prison id attribute. PRISONER have two foreign key in Volunteer id and Prison id, the primary key is Prisoner id. The CHILD entity have a foreign key in Prisoner id and the primary key is Child id. PERIOD entity don't have foreign key but his primary key is Period id. The last entity is POSITION, that has a primary key in the Position id attribute.


PERIOD
periodID
periodInitial
periodLast
POSITION
positionID
positionName
posDescription
posTerms
BOARD
boardID
periodID
volunteerID
positionID
VOLUNTEER
volunteerID
volunteerName
volunteerLastName
volunteerAddress
volunteerCity
volunteerZipCode
volunteerChurch
volunteerGender
volunteerPhone
volunteerOccupation
volunteerChaplain
volunteerInterest
volunteerZone
VISIT
volunteerID
prisonID
visitDay
visitHour
visitProyect
visitID
PRISON
prisonID
prisonName
prisonAddress
prisonCity
prisonZipCode
prisonGender
PRISONER
volunteerID
prisonID
prisonerID
prisonerName
prisonerGender
prisonerZone
CHILD
prisonerID
childName
childLastName
childAge
childGender
childAddress
childID

Physical Design:
Data dictionary:

- Entity type: Volunteer

Attributes	Attribute Data type	Required or Optional	Example Instance
volunteerName	varchar(20)	required	Melissa
volunteerLastName	varchar(20)	required	Kraft
volunteerAddress	varchar(30)	required	1422 Heft Ave
volunteerCity	varchar(20)	required	Miami
volunteerZipCode	varchar(5)	required	33321
volunteerChurch	varchar(40)	required	ID
volunteerGender	varchar(1)	required	F
volunteerPhone	varchar(10)	required	7877555555
volunteerOccupation	varchar(30)	optional	Student
volunteerChaplain	varchar(3)	required	Yes
volunteerInterest	varchar(30)	optional	
volunteerZone	varchar(1)	required	1

- Entity type: Period

Attributes	Attribute Data type	Required or Optional	Example Instance
periodInitial	varchar(10)	required	April
periodLast	varchar(10)	optional	

- Entity type: Prison

Attributes	Attribute Data type	Required or Optional	Example Instance
prisonName	varchar(15)	required	Oso Blanco
prisonAddress	varchar(30)	optional	
prisonCity	varchar(15)	required	San Juan
prisonZipCode	varchar(6)	required	00976
prisonGender	varchar(1)	required	M

- Entity type: Prisoner

Attributes	Attribute Data type	Required or Optional	Example Instance
prisonerName	varchar(15)	required	Juan
prisonID	int(10)	required	123
volunteerID	int(10)	required	82224
prisonerGender	varchar(1)	required	M
prisonerZone	varchar(1)	required	1

- Entity type: Child

Attributes	Attribute Data type	Required or Optional	Example Instance
■ prisonerID	int(10)	required	190009
■ childName	varchar(15)	required	Liz
■ childLastName	varchar(15)	required	Lopez
■ childAge	varchar(2)	required	9
■ childGender	varchar(1)	required	F
■ childAddress	varchar(20)	required	Carolina

- Entity type: Visit

Attributes	Attribute Data type	Required or Optional	Example Instance
■ volunteerID	int(10)	required	82224
■ prisonID	int(10)	required	123
■ visitDay	varchar(10)	required	Monday
■ visitHour	varchar(8)	required	7:00 pm
■ visitProgram	varchar(20)	required	Angel Tree

- Entity type: Board

Attributes	Attribute Data type	Required or Optional	Example Instance
■ volunteerID	int(10)	required	82224
■ periodID	int(10)	required	1516
■ positionID	int(10)	required	1

- Entity type: Position

Attributes	Attribute Data type	Required or Optional	Example Instance
■ positionName	varchar(15)	required	President
■ positionDescription	text(40)	optional	
■ positionTerms	varchar(1)	optional	2

Physical design decisions:

In the physical design decisions made we notice that we don't have to insert the id's in the tables because the engine create one id automatically in every insert we make, for that reason in the Data Dictionary we don't include the primary key of the respective table that we initially prognosticate, but we left the initial name id for the foreigns key. For example, the Child entity don't have childID attribute we mention in the conceptual and logical design, but have a prisonerID that is the foreign key of that entity.

In the Visit entity initially the combination of prison id and volunteer id was the primary in the logical design, but in the physical we notice that was better to use one unique id in the form than the combination of this two. For that reason we make a modification in the logical design to include the visitID as primary key and make volunteer id and prison id the foreign keys. Was in this step when we notice and start the research to make the surrogate primary key we mention before for the Child id.

Another decision was made because since we are making our database in sqlite and sqlite don't have a boolean class in their data type, we change the bool of the attribute volunteerChaplain to a varchar so the user can insert 'YES' or 'NO' respectively.

Index:

Create an index to a non-primary key has its benefits when you make a lot of searches with that non-primary key because make the search more fast. But you have to analyse is you have the capacity to handle more space in the CPU, if your tables are too big or not because can get an computational overhead. In our case if were going to create an index on a non-primary key field, would be the Zone attributes of the Volunteer and Prisoner tables. Because that attributes, prisonerZone and volunteerZone, eventually could be searched in many ways for statistics. For example, from where come the most people that is now in jail? How we can help to impact positively that community to reduce the criminality?

Database Build:

CREATE TABLE Volunteer (id integer primary key, volunteerName varchar(20) not null, volunteerLastName varchar(20) not null, volunteerAddress varchar(30) not null, volunteerCity varchar(20) not null, volunteerZipCode varchar(5) not null, volunteerChurch varchar(40) not null, volunteerGender varchar(1) not null, volunteerPhone varchar(10) not null, volunteerOccupation varchar(30), volunteerChaplain varchar(3) not null, volunteerInterest varchar(30), volunteerZone varchar(1) not null DEFAULT '1');

CREATE TABLE Prison (id integer primary key, prisonName varchar(15) not null, prisonAddress varchar(30), prisonCity varchar(15) not null, prisonZipCode int(6) not null, prisonGender varchar(1) not null);

CREATE TABLE Prisoner(id integer primary key, prisonerName varchar(15) not null, prisonerGender varchar(1) not null, prisonerZone varchar(1), prisonID int(10) not null, volunteerID int(10) not null, foreign key (prisonID) references Prison(id) on update cascade, foreign key (volunteerID) references Volunteer(id) on update cascade, check(volunteerID in (select id from Volunteer)), check(prisonID in (select id from Prison)));

CREATE TABLE Visit(id integer primary key, volunteerID int not null, prisonID int not null, visitDay varchar(10) not null, visitHour varchar(8) not null, visitProgram varchar(20) not null, foreign key (volunteerID) references Volunteer(id) on delete cascade, foreign key (prisonID) references Prison(id) on update cascade);

CREATE TABLE Period(id integer primary key, periodInitial varchar(10) not null, periodLast varchar(10));

CREATE TABLE Child(id integer primary key, prisonerID int, childName varchar(15), childLastName varchar(15), childAge varchar(2) not null, childGender varchar(1) not null, childAddress varchar(20) not null, foreign key (prisonerID) references Prisoner(id) on update cascade);

CREATE TABLE Position(id integer primary key, positionName varchar(15) check(positionName is not null), posDescription text, posTerms int(1) not null);

CREATE TABLE Board(id integer primary key, periodID int, volunteerID int, positionID int, foreign key (positionID) references `Position`(id) on update cascade, foreign key (volunteerID) references Volunteer(id) on update cascade, foreign key (periodID) references Period(id) on update cascade, check(volunteerID in (select id from Volunteer)));

Enter Data:

INSERT INTO Volunteer (volunteerName, volunteerLastName, volunteerAddress, volunteerCity, volunteerZipCode, volunteerChurch, volunteerGender, volunteerPhone, volunteerOccupation, volunteerChaplain, volunteerInterest, volunteerZone) VALUES ('Jon', 'Snow', 'Urb. Wall', 'Isabela', '00906', 'MB', 'M', '9398532298', 'Guardia', 'Si', 'Enseñar', 2);

INSERT INTO Volunteer (volunteerName, volunteerLastName, volunteerAddress, volunteerCity, volunteerZipCode, volunteerChurch, volunteerGender, volunteerPhone, volunteerOccupation, volunteerChaplain, volunteerInterest, volunteerZone) VALUES ('Bianca', 'Colon', 'Urb. Sky', 'San Juan', '00926', 'MII', 'F', '7877555555', 'estudiante', 'Si', 'leer', 1);

INSERT INTO Volunteer (volunteerName, volunteerLastName, volunteerAddress, volunteerCity, volunteerZipCode, volunteerChurch, volunteerGender, volunteerPhone, volunteerOccupation, volunteerChaplain, volunteerInterest, volunteerZone) VALUES ('Marta', 'Cortez', 'Urb. Palo Alto', 'Ponce', '00896', 'Protestante', 'F', '7877534569', 'psicologa', 'Si', 'arbol angel', 3);

INSERT INTO Volunteer (volunteerName, volunteerLastName, volunteerAddress, volunteerCity, volunteerZipCode, volunteerChurch, volunteerGender, volunteerPhone, volunteerOccupation, volunteerChaplain, volunteerInterest, volunteerZone) VALUES ('Abi', 'Ayala', 'Urb. Castellana', 'Carolina', '00926', 'Catolica', 'M', '9398588288', 'doctor', 'No', 'operar', 6);

INSERT INTO Volunteer (volunteerName, volunteerLastName, volunteerAddress, volunteerCity, volunteerZipCode, volunteerChurch, volunteerGender, volunteerPhone, volunteerOccupation, volunteerChaplain, volunteerInterest, volunteerZone) VALUES ('Vivian', 'Ayala', '356 Suri', 'Humacao', '00836', 'MI', 'F', '3568588288', 'Programadora', 'No', 'Paginas Web', 4);

INSERT INTO Prison (prisonName, prisonAddress, prisonCity, prisonZipCode, prisonGender) VALUES ('Mostro Verde', 'Ponce Mil', 'Ponce', '00978', 'M');

```
INSERT INTO Prison (prisonName, prisonAddress, prisonCity, prisonZipCode, prisonGender)
VALUES ('Mostra Roja', 'Ponce Mil', 'Ponce', '00978', 'F');

INSERT INTO Prison (prisonName, prisonAddress, prisonCity, prisonZipCode, prisonGender)
VALUES ('1072', 'Calle Casio', 'Bayamon', '00976', 'M');

INSERT INTO Prison (prisonName, prisonAddress, prisonCity, prisonZipCode, prisonGender)
VALUES ('500', 'Calle Casio', 'Bayamon', '00976', 'F');

INSERT INTO Prison (prisonName, prisonAddress, prisonCity, prisonZipCode, prisonGender)
VALUES ('Bayamon 483', 'Calle Casio', 'Bayamon', '00976', 'M');

INSERT INTO Visit (volunteerID, prisonID, visitDay, visitHour, visitProgram) VALUES (1,3,
'martes', '7:00 pm', 'Angel Tree');

INSERT INTO Visit (volunteerID, prisonID, visitDay, visitHour, visitProgram) VALUES (1,4,
'sabados', '10:00 am', 'Discipulado') ;

INSERT INTO Visit (volunteerID, prisonID, visitDay, visitHour, visitProgram) VALUES (2,4,
'LMW', '5:00 pm', 'Angel Tree');

INSERT INTO Visit (volunteerID, prisonID, visitDay, visitHour, visitProgram) VALUES (3,4,
'LMW', '5:00 pm', 'Angel Tree');

INSERT INTO Visit (volunteerID, prisonID, visitDay, visitHour, visitProgram) VALUES (5,5, 'J',
'5:00 pm', 'Angel Tree');

INSERT INTO Prisoner (prisonerName, prisonerGender, prisonerZone, prisonID, volunteerID)
VALUES ('Juan', 'M', 1, 2, 5);

INSERT INTO Prisoner (prisonerName, prisonerGender, prisonerZone, prisonID, volunteerID)
VALUES ('Maria', 'F', 2, 3, 3);

INSERT INTO Prisoner (prisonerName, prisonerGender, prisonerZone, prisonID, volunteerID)
VALUES ('Marcos', 'M', 5, 4, 2);

INSERT INTO Prisoner (prisonerName, prisonerGender, prisonerZone, prisonID, volunteerID)
VALUES ('Rafael', 'M', 4, 4, 1);

INSERT INTO Prisoner (prisonerName, prisonerGender, prisonerZone, prisonID, volunteerID)
VALUES ('Miguelina', 'F', 4, 5, 1);
```

```
INSERT INTO Child (prisonerID, childName, childLastName, childAge, childGender,
childAddress) VALUES (3, 'Liz', 'Lopez', 5, 'F', 'Florida');

INSERT INTO Child (prisonerID, childName, childLastName, childAge, childGender,
childAddress) VALUES (4, 'Samantha', 'Rosado', 2, 'F', 'Florida');

INSERT INTO Child (prisonerID, childName, childLastName, childAge, childGender,
childAddress) VALUES (2, 'Andrea', 'Rosario', 15, 'F', 'Carolina');

INSERT INTO Child (prisonerID, childName, childLastName, childAge, childGender,
childAddress) VALUES (1, 'Juan', 'Lopez', 7, 'M', 'Fajardo');

INSERT INTO Child (prisonerID, childName, childLastName, childAge, childGender,
childAddress) VALUES (1, 'Pedro', 'Lopez', 11, 'M', 'Luquillo');

INSERT INTO Position (positionName, posDescription, posTerms) VALUES ('Presidente',
'presentar', 2);

INSERT INTO Position (positionName, posDescription, posTerms) VALUES ('vice-presidente',
'presentar si falta', 2);

INSERT INTO Position (positionName, posDescription, posTerms) VALUES ('secretario',
'apuntar', 3);

INSERT INTO Position (positionName, posDescription, posTerms) VALUES ('SubSecretario',
'presentar', 2);

INSERT INTO Position (positionName, posDescription, posTerms) VALUES ('Tesorero',
'contar', 2);

INSERT INTO Period (periodInitial, periodLast) VALUES ('april 2015', null);

INSERT INTO Period (periodInitial, periodLast) VALUES ('april 2014', 'march 2015');

INSERT INTO Period (periodInitial, periodLast) VALUES ('dic 2012', 'dic 2013');

INSERT INTO Period (periodInitial, periodLast) VALUES ('oct 2011', 'sep 2012');

INSERT INTO Period (periodInitial, periodLast) VALUES ('june', null);

INSERT INTO Board (periodID, volunteerID, positionID) VALUES (1, 1, 1);

INSERT INTO Board (periodID, volunteerID, positionID) VALUES (1, 2, 2);
```

```
INSERT INTO Board (periodID, volunteerID, positionID) VALUES (2, 3, 4);

INSERT INTO Board (periodID, volunteerID, positionID) VALUES (2, 5, 2);

INSERT INTO Board (periodID, volunteerID, positionID) VALUES (5, 5, 5);
```

Critical Analysis:

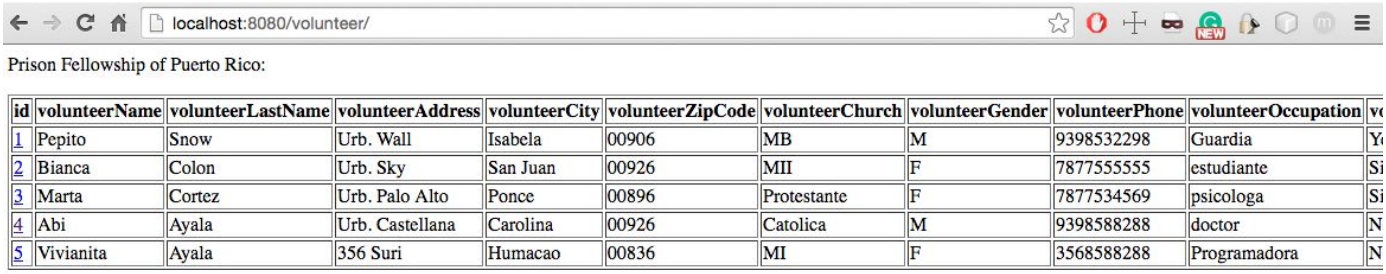
We made some changes in the Attributes Data types, for example in the address zip code, because initially I've haved as int, because the zipcode is compounded by numbers but when we insert 00926 as a Puerto Rico normal zipcode then when we see the database we found a 926. For that reason I have to change it from int to a varchar. We also increment the size of the address to varchar(30) for the cases where the address is long. For the phone number we need to increment the size for the cases that the phone number include +1-787-755-555. Other changes was made and commented during the process.

Application (user interface):

The application user interface has been created using Bottle, a lightweight web-framework for Python. Bottle use sqlite3 as a database. The application interface start after we run: python db.py to create the database and python ccpr.py. After run the ccpr.py we can access the port 8080 from the browser to see our application in /ccpr/.

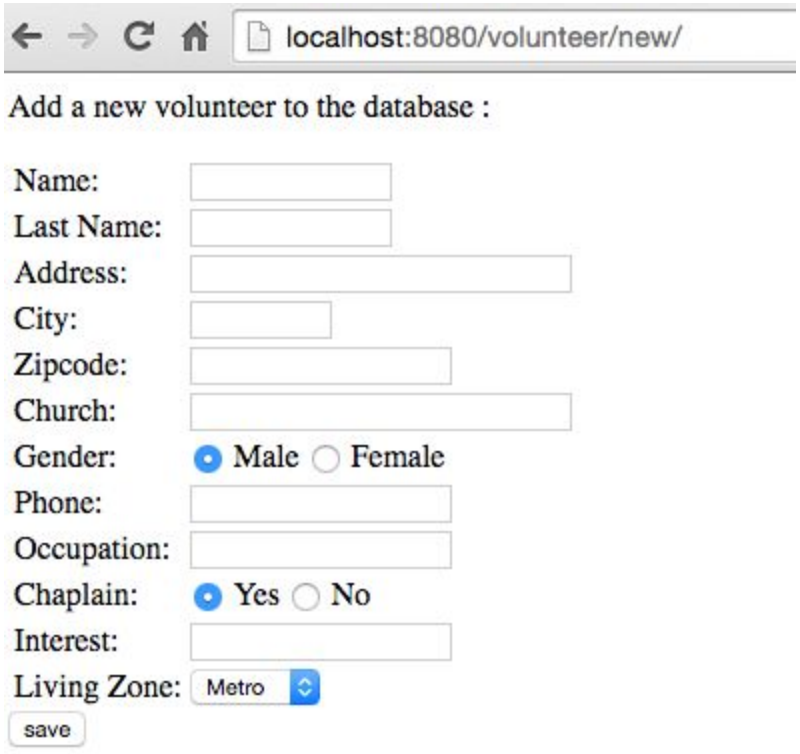
There, we can select links to view/insert to specific tables.

For example, clicking the List link of Volunteer, we obtain all the content in the Volunteer table, also we can see this table in /volunteer/

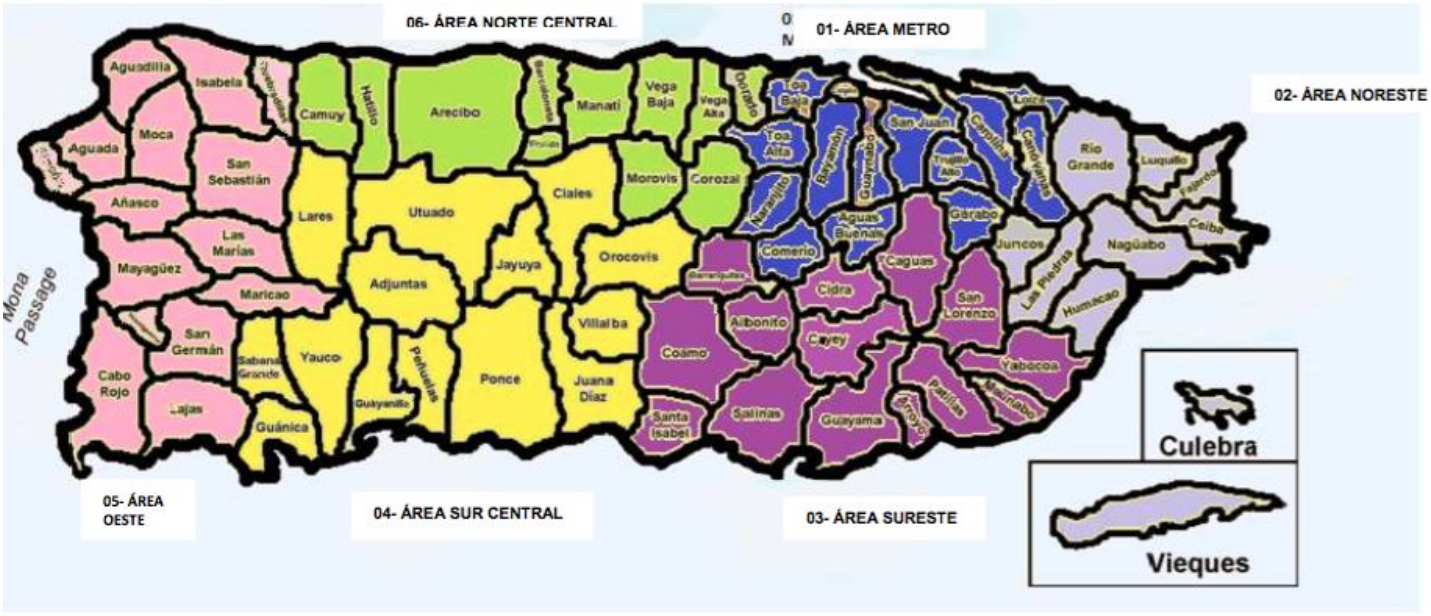


id	volunteerName	volunteerLastName	volunteerAddress	volunteerCity	volunteerZipCode	volunteerChurch	volunteerGender	volunteerPhone	volunteerOccupation	v
1	Pepito	Snow	Urb. Wall	Isabela	00906	MB	M	9398532298	Guardia	Y
2	Bianca	Colon	Urb. Sky	San Juan	00926	MII	F	7877555555	estudiante	Si
3	Marta	Cortez	Urb. Palo Alto	Ponce	00896	Protestante	F	7877534569	psicologa	Si
4	Abi	Ayala	Urb. Castellana	Carolina	00926	Catolica	M	9398588288	doctor	N
5	Vivianita	Ayala	356 Suri	Humacao	00836	MI	F	3568588288	Programadora	N

If we click the Insert link of Volunteer, we will get a form to insert the volunteers information, that form is saved in /volunteer/new/.



The Living Zone would be selected following the zones assigned in the map provided by the organization. This map would be included in the right side of the form.



Also we can create our specific queries.

To search for especific information:

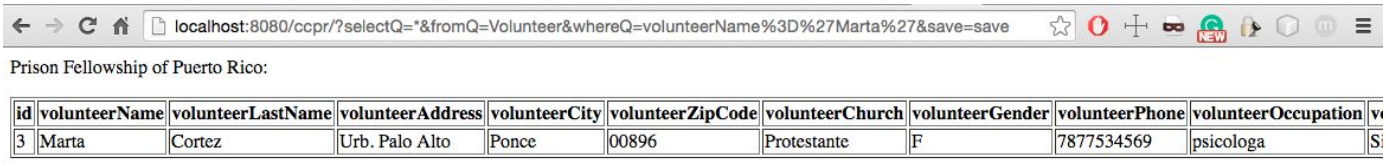
SELECT *

FROM Volunteer

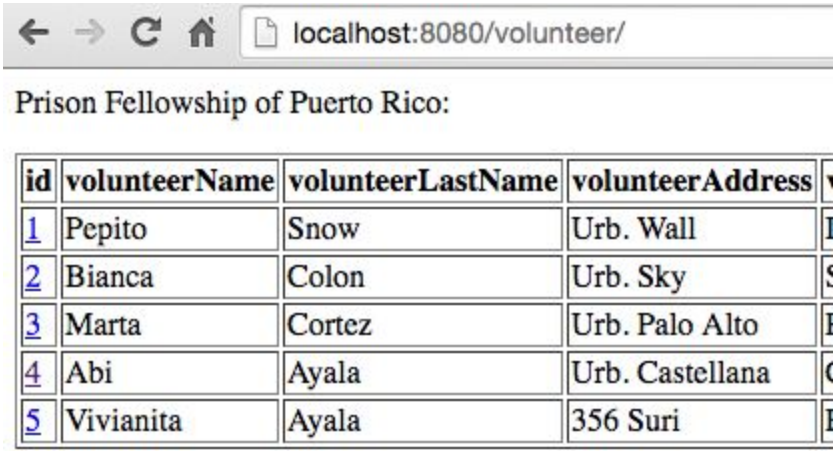
WHERE volunteerName='Marta'

save

And see the results.



To update some content, we can access the id of the row we want to edit in the List link. There you can notice that the id column is linked to other page.

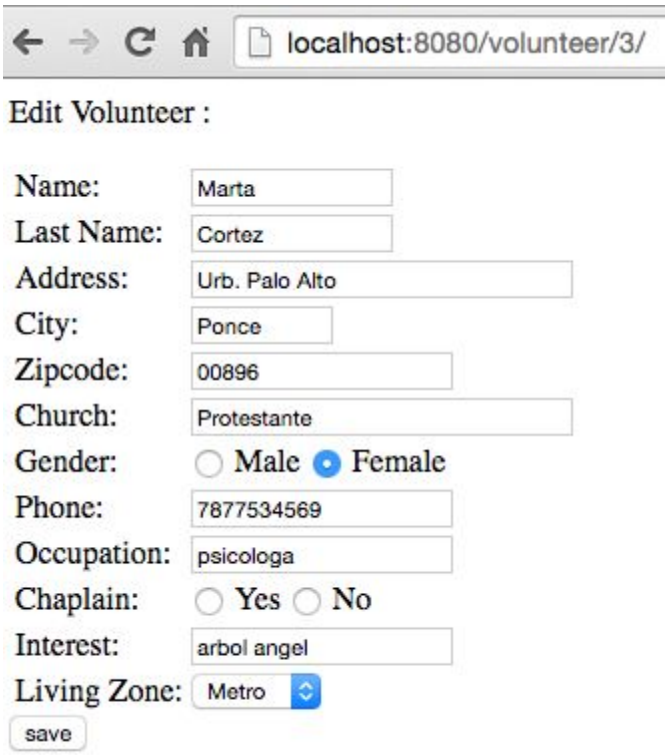


id	volunteerName	volunteerLastName	volunteerAddress
1	Pepito	Snow	Urb. Wall
2	Bianca	Colon	Urb. Sky
3	Marta	Cortez	Urb. Palo Alto
4	Abi	Ayala	Urb. Castellana
5	Vivianita	Ayala	356 Suri

id	volunteerName
1	Pepito
2	Bianca
3	Marta
4	Abi
5	Vivianita

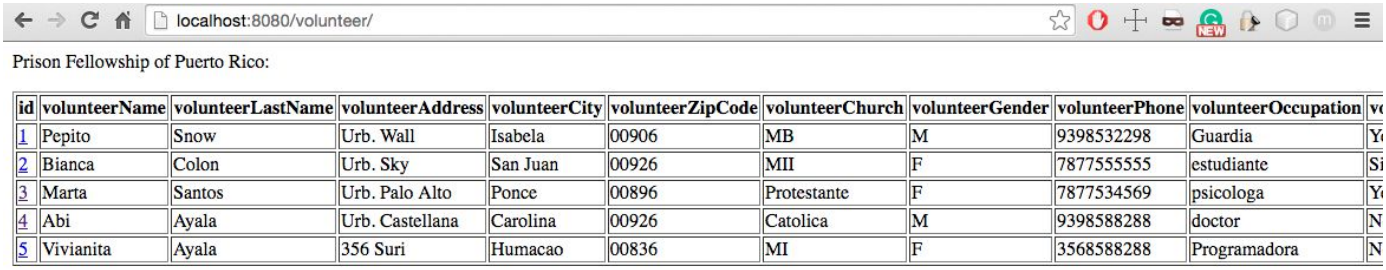
If we click the id number 3, that is Marta we can access to her information and make the changes we want. Click the save button and the changes will be saved in the same id.

BEFORE



AFTER

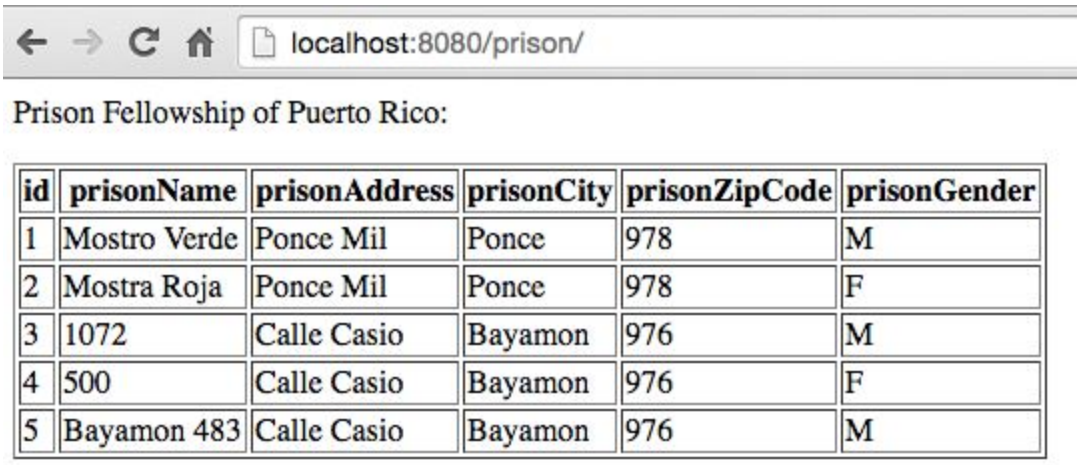
And if we run the specific query or list all the table we can see the update.



id	volunteerName	volunteerLastName	volunteerAddress	volunteerCity	volunteerZipCode	volunteerChurch	volunteerGender	volunteerPhone	volunteerOccupation	v
1	Pepito	Snow	Urb. Wall	Isabela	00906	MB	M	9398532298	Guardia	Y
2	Bianca	Colon	Urb. Sky	San Juan	00926	MII	F	7877555555	estudiante	Si
3	Marta	Santos	Urb. Palo Alto	Ponce	00896	Protestante	F	7877534569	psicologa	Y
4	Abi	Ayala	Urb. Castellana	Carolina	00926	Catolica	M	9398588288	doctor	N
5	Vivianita	Ayala	356 Suri	Humacao	00836	MI	F	3568588288	Programadora	N

id	volunteerName	volunteerLastName	volunteerAddress	volunteerCity	volunteerZipCode	volunteerChurch	volunteerGender	volunteerPhone	volunteerOccupation	v
3	Marta	Santos	Urb. Palo Alto	Ponce	00896	Protestante	F	7877534569	psicologa	Y

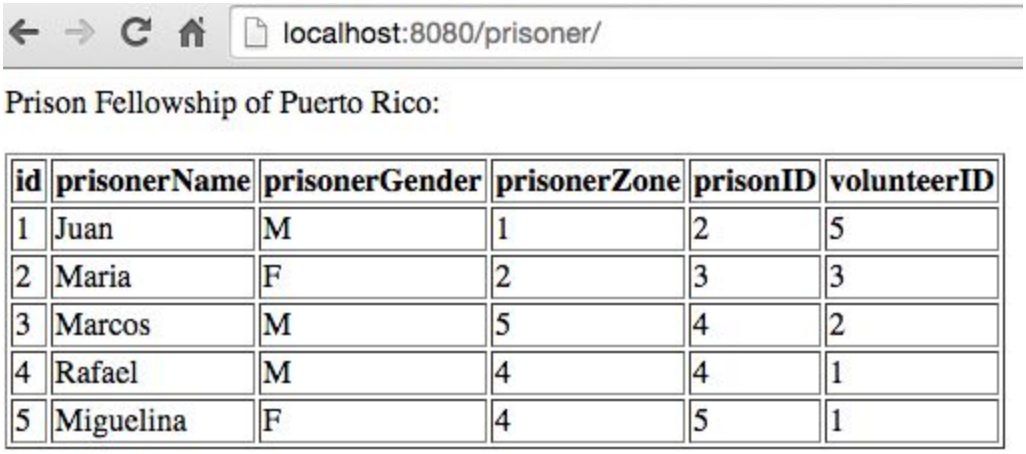
The prison table can be accessed in /prison/.



id	prisonName	prisonAddress	prisonCity	prisonZipCode	prisonGender
1	Mostro Verde	Ponce Mil	Ponce	978	M
2	Mostra Roja	Ponce Mil	Ponce	978	F
3	1072	Calle Casio	Bayamon	976	M
4	500	Calle Casio	Bayamon	976	F
5	Bayamon 483	Calle Casio	Bayamon	976	M

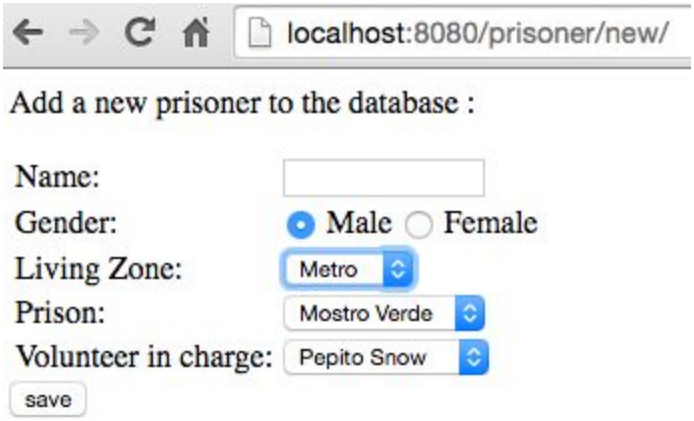
And the form is in /prison/new/

The table Prisoner will be accessed in /prisoner/

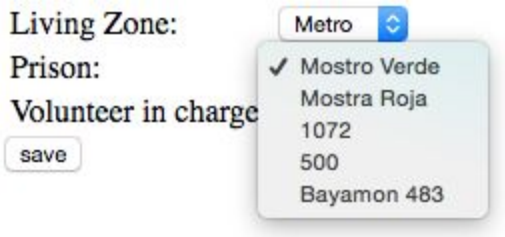


id	prisonerName	prisonerGender	prisonerZone	prisonID	volunteerID
1	Juan	M	1	2	5
2	Maria	F	2	3	3
3	Marcos	M	5	4	2
4	Rafael	M	4	4	1
5	Miguelina	F	4	5	1

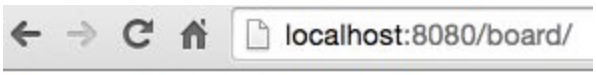
And the form to insert is in /prisoner/new/. There you can select the prison and the volunteer. This two take the information from their respective table and display their contect.



This are the prisons saved in Prison table:



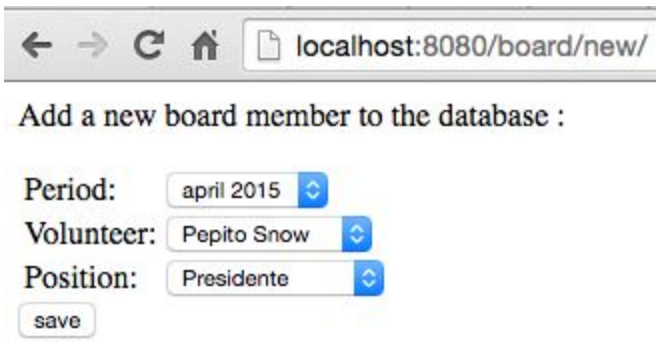
The Board table just contain the ids,



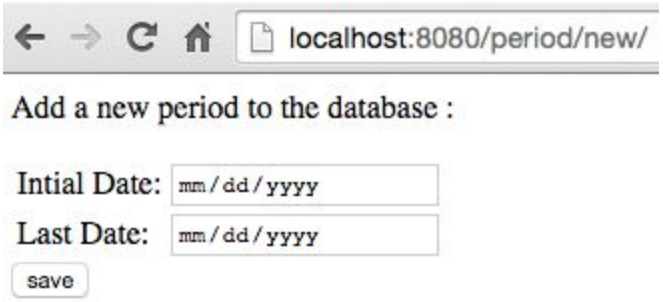
Prison Fellowship of Puerto Rico:

id	periodID	volunteerID	positionID
1	1	1	1
6	1	1	1
2	1	2	2
4	2	5	2
3	2	3	4
5	5	5	5

but when we want to insert, the form go to the respective table and get the information. In that way the user just have to select by name and not by number.



Period Form:

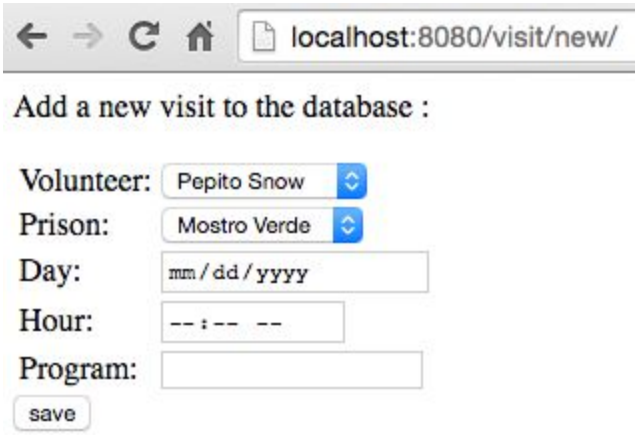


Visit Table:

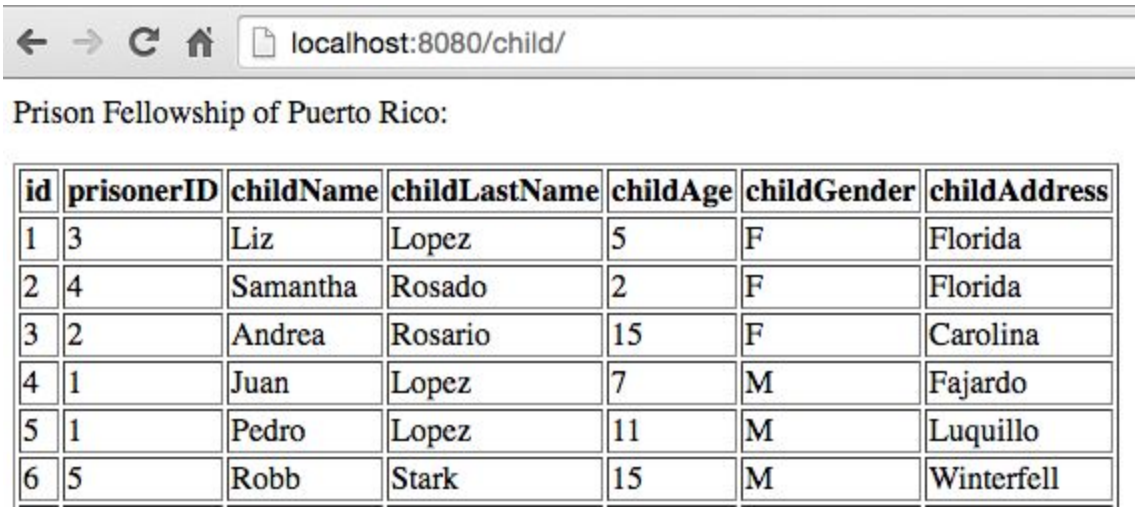
Prison Fellowship of Puerto Rico:

id	volunteerID	prisonID	visitDay	visitHour	visitProgram
1	1	3	martes	7:00 pm	Angel Tree
2	1	4	sabados	10:00 am	Discipulado
3	2	4	LMW	5:00 pm	Angel Tree
4	3	4	LMW	5:00 pm	Angel Tree
5	5	5	J	5:00 pm	Angel Tree
6	2	4	2016-05-11	12:59	Solicitudes

Visit Form:

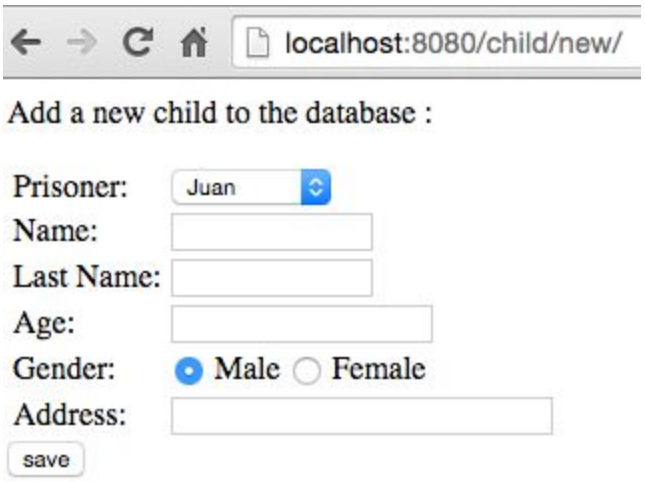


Child table:

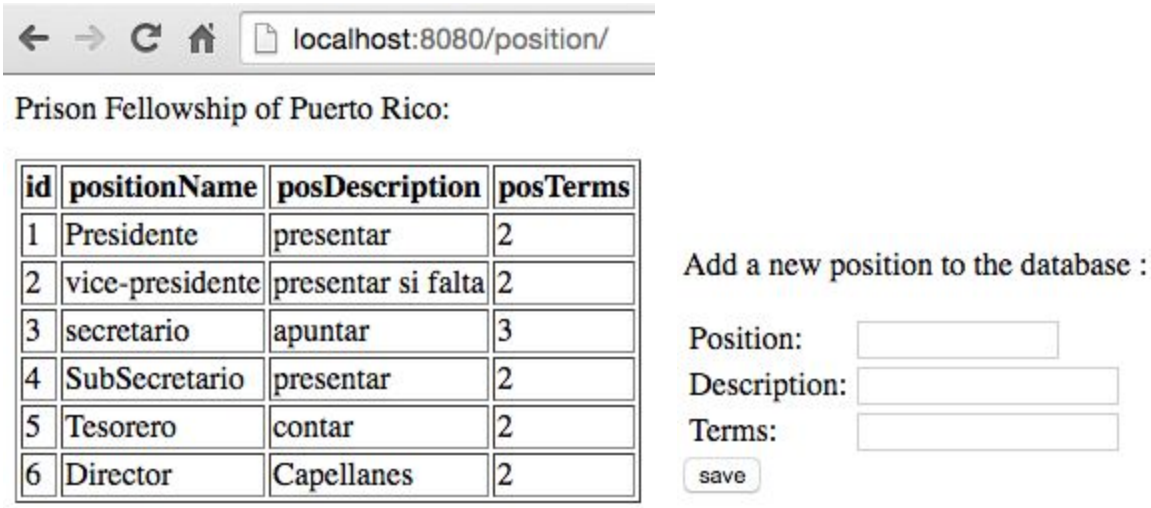


id	prisonerID	childName	childLastName	childAge	childGender	childAddress
1	3	Liz	Lopez	5	F	Florida
2	4	Samantha	Rosado	2	F	Florida
3	2	Andrea	Rosario	15	F	Carolina
4	1	Juan	Lopez	7	M	Fajardo
5	1	Pedro	Lopez	11	M	Luquillo
6	5	Robb	Stark	15	M	Winterfell

Child Form:



Position table and form in /position/new/:



id	positionName	posDescription	posTerms
1	Presidente	presentar	2
2	vice-presidente	presentar si falta	2
3	secretario	apuntar	3
4	SubSecretario	presentar	2
5	Tesorero	contar	2
6	Director	Capellanes	2

Queries:

Query Name: Board Members
User Purpose: I want to find the boards of past years. This query shows the period initial day, position and names of all board members in database.
SQL statement:

```
Select periodInitial, positionName, volunteerName, volunteerLastName
From Board, Volunteer, Period,Position
Where Board.volunteerID= Volunteer.id and
Board.periodID= Period.id and Board.positionID = Position.id Order by periodInitial asc
```

periodInitial	positionName	volunteerName	volunteerLastName
april 2014	SubSecretario	Marta	Cortez
april 2014	vice-presidente	Vivian	Ayala
april 2015	Presidente	Jon	Snow
april 2015	vice-presidente	Bianca	Colon
june	Tesorero	Vivian	Ayala

--

Query Name: Volunteers visiting prisons
User Purpose: This are the volunteers that visit prisons. This query shows the volunteers name and last name, the prison they visit, day and hours.
SQL statement:

```
Select volunteerName, VolunteerLastName, Prison.prisonName, visitDay, visitHour
from Volunteer, Prison, Visit
Where Visit.volunteerID = Volunteer.id and Visit.prisonID = Prison.id
Order by volunteerName, PrisonName
```

volunteerName	volunteerLastName	prisonName	visitDay	visitHour
Bianca	Colon	500	LMW	5:00 pm
Jon	Snow	1072	martes	7:00 pm
Jon	Snow	500	sabados	10:00 am
Marta	Cortez	500	LMW	5:00 pm
Vivian	Ayala	Bayamon 483	J	5:00 pm

--

Query Name: How many chaplains
User Purpose: With the number of chaplains they can estimate the monetary income for the next year. Assuming this 3 chaplains pay the $50 that cost the id renovation, then the next year they could get $150.
SQL statement:

```
Select count(*) as Chaplains, count(*)*50 as EstimateIncomeForNextYear
From Volunteer
Where volunteerChaplain = 'Si'
```

Chaplains	EstimateIncomeForNextYear
3	150

--

Query Name: How many womens and mens
User Purpose: Count the volunteers womens and mens in the organization.
SQL statement:

```
Select volunteerGender, count(volunteerGender) as total
From Volunteer
group by volunteerGender
```

volunteerGender	total
F	3
M	2

Remember:

We are not providing full data validation, but we implement different ways of data validation in the more critical cases. Also in the application interface the forms has specific validation for the critical cases. The data validation helps to promote data integrity because helps to maintain the data uniform, is a value is supposed to be an integer but because we don't have data validation we can insert a letter and then when we make a search by integers we are losing the data obtained in that value or worst the database broke or get corrupted. For that reason in a next version of this database we want to include full data validation and error handling. Also we want to get a better user interface with more features, to make it look attractive to the user.