



CPCA

Database Documentation 2017

Carson Badame, Marcos Barbieri, Jimmy Crowley, Jesse Opitz, John Randis, Rachel Ulicni

---

## *Executive Summary*

---

This document's objective and purpose is to provide a design for and implementation of a database for The Center for the Prevention of Child Abuse of Dutchess County, as well as essential information regarding the architecture of the database. The CPCA requires a database to be able to store, insert, update, delete, and otherwise manipulate data, as well as view reports of data, for their business records and purposes. The database will be key in taking attendance for classes, building and viewing reports on participants, and storing class survey data.

All of the CPCA's data which was previously recorded on plain paper and stored in filing cabinets will now be accounted for in the database, and can quickly and easily be found, rather than manually storing and searching through physical pieces of paper for information. The database will be integrated with the client application, which will provide a clear front-end that will allow the users at the CPCA to easily and effectively retrieve data from the database. The implementation of the database will coincide with the use of physical paper reporting, until the system is proven effective and the users obtain a level of comfortability with the application utilizing the database, ultimately eliminating the need to use paper data storage and recovery.

The design encompasses all aspects of the CPCA's data retrieval, including—but not limited to—details about participants, classes, employees, reports, surveys, referrals and families. This database will only be accessible from the CPCA's secure network at its headquarters due to the high level of sensitive and confidential information that it contains.

## Entity Relationship Diagram

An Entity Relationship Diagram, also known as an ER diagram or ERD, is a data modeling technique that graphically illustrates an information system's entities and the relationship between those entities. The elements of an ERD are:

- **Entities**
- **Attributes**
- **Relationships (Cardinality)**

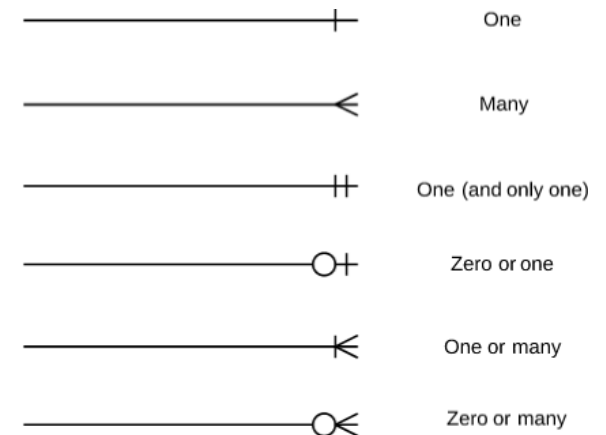
An **entity** is a real-world item or concept that exists on its own, typically a noun—such as a person, object, concept, or event—that can have data stored about it. It is represented by the tables in the database. Each row of the table is an instance of that entity. For example, a customer, student, car, or product.

An **attribute** of an entity is a particular property that describes the entity. For example, attributes for a student entity may be: student\_id, first\_name, last\_name, email, phone\_number.

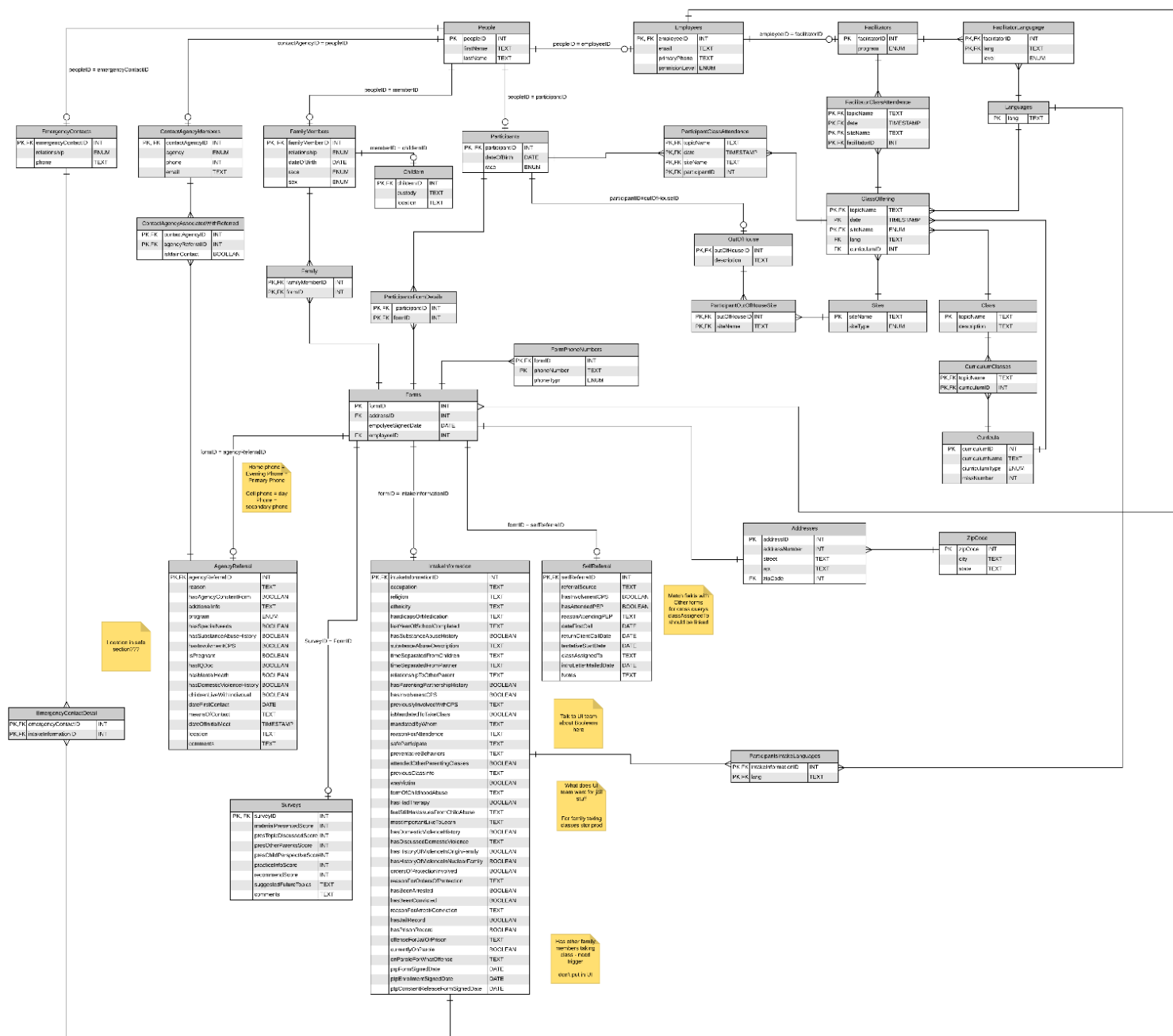
A **relationship** is the association that describes the interaction between entities, like a verb. For example, if a student registers for a course, the two entities would be the student and the course, and the relationship depicted is the act of enrolling, connecting the two entities in that way. Relationships are typically shown as connecting lines.

**Cardinality** defines relationships in terms of numbers; it is the number of instances that one entity can, or must, be associated with each instance of another entity. This can be **one-to-one**, **one-to-many**, or **many-to-many** relationships.

- **One-to-One:** Both tables can have only one record on either side of the relationship.
  - For example, one student associated with one mailing address.
- **One-to-Many:** The first table contains only one record that relates to none, one, or many records in the second, related table.
  - For example, one student registers for multiple courses, but all those courses have a single line back to that one student.
- **Many-to-Many:** Each record in one table can relate to any number of records (or no records) in the other table.
  - Students as a group are associated with multiple faculty members, and faculty members in turn are associated with multiple students.



## CPCA Database Documentation



---

## Primary and Foreign Keys

---

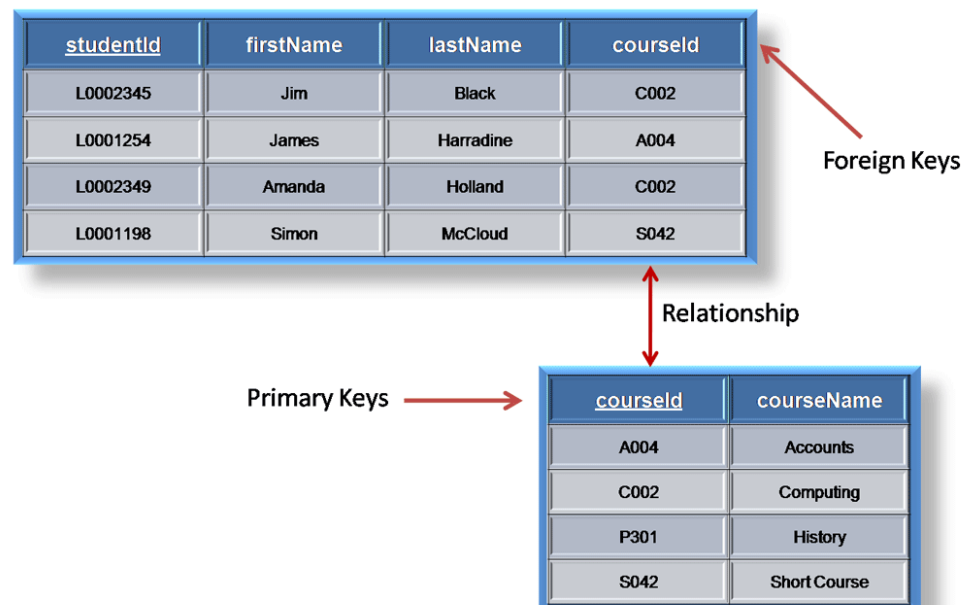
Primary keys and foreign keys are extremely important concepts and are critical to an efficient relational database. Without them, relational databases would not work. A **primary key** (PK) is a column, or columns, in a table that uniquely identifies the rows in that table. In order for a table to qualify as a relational table, it must have a primary key. The primary key's main features are:

- It must contain a unique value for each row of data.
- It cannot contain null values.

For example, the primary key in a *Students* table may be *student\_id*, since that is a unique number which identifies a specific student.

A **foreign key** (FK) is a column, or columns, in a table that refers to the primary key in another table. Unlike primary keys, duplicate and null values are allowed in foreign keys. Foreign keys allow for *referential integrity*, meaning that if a foreign key contains a value, this value refers to an existing record in the related table.

In the example below, the *Courses* table has a primary key of *courseId*, which is a foreign key in the *Students* table. The cardinality of their relationship is *many-to-many*, because a student can have many courses, and a course can have many students.

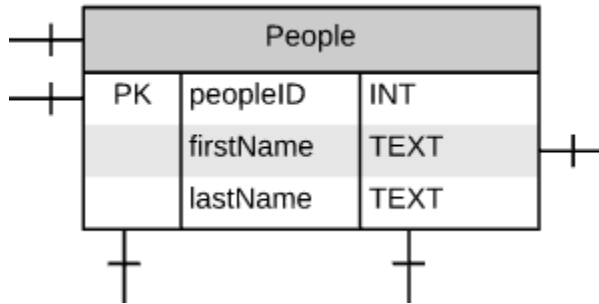


---

## Tables

---

### People Table



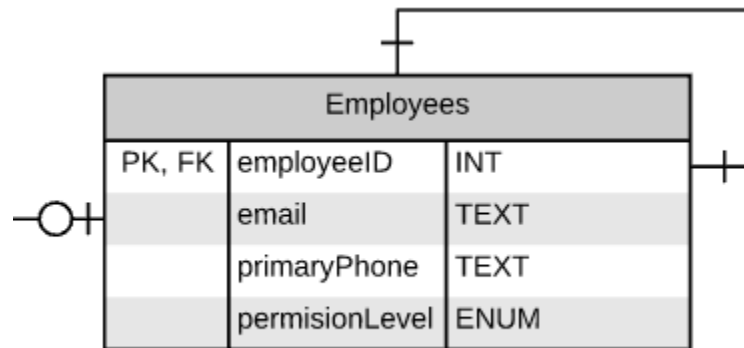
The People Table stores general data regarding any person. This table has the following attributes:

- peopleID (PK)
- firstName
- lastName

The People Table has 5 relationships and they are:

Table	Relationship	Other Table
People	One-to-Zero or One	EmergencyContacts
People	One-to-Zero or One	ContactAgencyMembers
People	One-to-Zero or One	FamilyMembers
People	One-to-Zero or One	Participants
People	One-to-Zero or One	Employees

## Employees Table



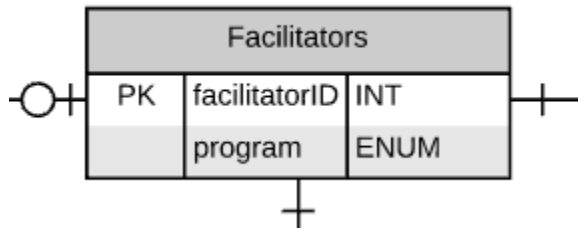
The Employees Table stores an employee's general data. The table has the following attributes:

- employeeID (PK/FK)
- email
- primaryPhone
- permissionLevel

The Employees Table has 3 relationships and they are:

Table	Relationship	Other Table
Employees	Zero or One-to-One	People
Employees	One-to-Zero or One	Facilitators
Employees	One-to-Many	Forms

## Facilitators Table



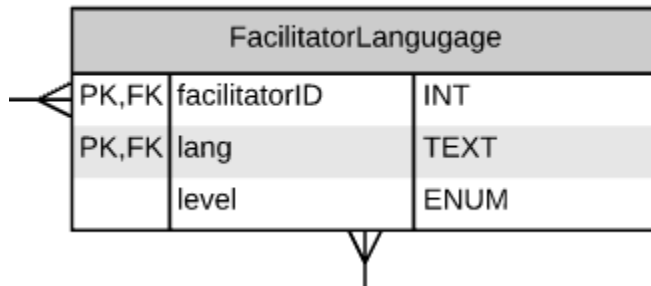
The Facilitators Table stores specific data for a facilitator type of employee. The table has the following attributes:

- facilitatorID (PK)
- program

The Facilitators Table has 3 relationships and they are:

Table	Relationship	Other Table
Facilitators	Zero or One-to-One	Employees
Facilitators	One-to-Many	FacilitatorClassAttendance
Facilitators	One-to-Many	FacilitatorLanguage

## FacilitatorLanguage Table



The FacilitatorLanguage Table stores data regarding the languages that a facilitator is able to speak. The table has the following attributes:

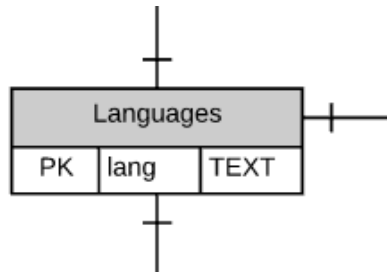
- facilitatorID (PK/FK)
- lang (PK/FK)
- level

The FacilitatorLanguage Table has 2 relationships and they are:

Table	Relationship	Other Table
FacilitatorLanguage	Many-to-One	Facilitators
FacilitatorLanguage	Many-to-One	Languages



## Languages Table



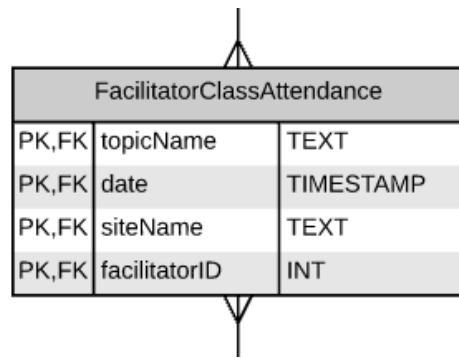
The Languages Table stores various languages. The table has the following attributes:

- lang (PK)

The Languages Table has 3 relationships and they are:

Table	Relationship	Other Table
Languages	One-to-Many	FacilitatorLanguage
Languages	One-to-many	ClassOffering
Languages	One-to-Many	ParticipantsIntakeLanguages

## FacilitatorClassAttendance Table



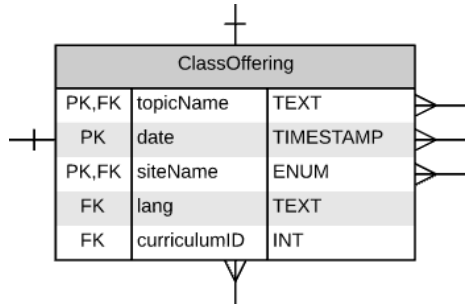
The FacilitatorClassAttendance Table stores data regarding class attendance taken by the facilitator. The table has the following attributes:

- topicName (PK/FK)
- date (PK/FK)
- siteName (PK/FK)
- facilitatorID (PK/FK)

The FacilitatorClassAttendance Table has 2 relationships and they are:

Table	Relationship	Other Table
FacilitatorClassAttendance	Many-to-One	Facilitators
FacilitatorClassAttendance	Many-to-One	ClassOffering

## ClassOffering Table



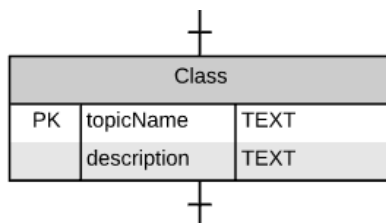
The ClassOffering Table stores data regarding class attendance taken by the facilitator. The table has the following attributes:

- topicName (PK/FK)
- date (PK)
- siteName (PK/FK)
- lang (FK)
- curriculumID (FK)

The ClassOffering Table has 6 relationships and they are:

Table	Relationship	Other Table
ClassOffering	Many-to-One	Languages
ClassOffering	Many-to-One	Curricula
ClassOffering	Many-to-One	Class
ClassOffering	Many-to-One	Sites
ClassOffering	One-to-Many	FacilitatorClassAttendance
ClassOffering	One-to-Many	ParticipantClassAttendance

## Class Table



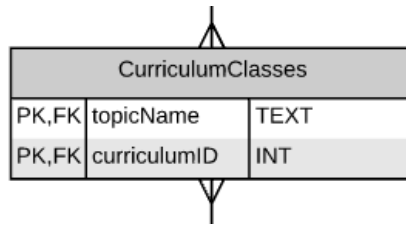
The Class Table stores basic data about classes. The table has the following attributes:

- topicName (PK)
- description

The Class Table has 2 relationships and they are:

Table	Relationship	Other Table
Class	One-to-Many	ClassOffering
Class	One-to-many	CurriculumClasses

## CurriculumClasses Table



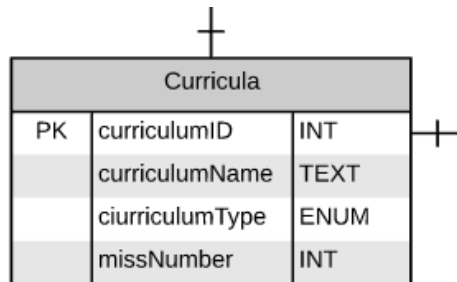
The CurriculumClass Table stores data connecting a class with a specific curriculum. The table has the following attributes:

- topicName (PK/FK)
- curriculumID (PK/FK)

The CurriculumClasses Table has 2 relationships and they are:

Table	Relationship	Other Table
CurriculumClass	Many-to-One	Class
CurriculumClass	Many-to-One	Curricula

## Curricula Table



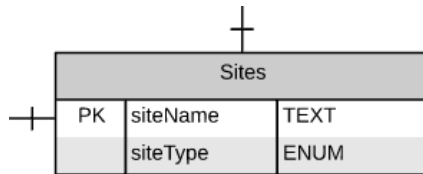
The Curricula Table stores data about specific curricula. The table has the following attributes:

- curriculumID (PK)
- curriculumName
- curriculumType
- missNumber

The Curricula Table has 2 relationships and they are:

Table	Relationship	Other Table
Curricula	One-to-Many	CurriculumClasses
ClassOffering	One-to-Many	ClassOffering

## Sites Table



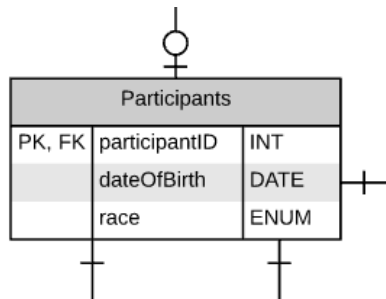
The Sites Table stores data about the type of various sites. The table has the following attributes:

- siteName (PK)
- siteType

The Sites Table has 2 relationships and they are:

Table	Relationship	Other Table
Sites	One-to-Many	ClassOffering
Sites	One-to-Many	ParticipantOutOfHouseSite

## Participants Table



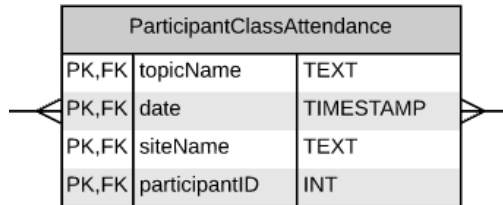
The Participants Table stores data about participants. The table has the following attributes:

- participantID (PK/FK)
- dateOfBirth
- race

The Participants Table has 4 relationships and they are:

Table	Relationship	Other Table
Participants	Zero or One-to-One	People
Participants	One-to-Many	ParticipantClassAttendance
Participants	One-to-Zero or One	OutOfHouse
Participants	One-to-Many	ParticipantsFormDetails

## ParticipantClassAttendance Table



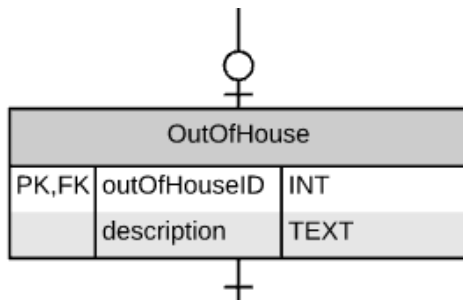
The ParticipantClassAttendance Table stores data about attendance as recorded by participants. The table has the following attributes:

- topicName (PK/FK)
- date (PK/FK)
- siteName (PK/FK)
- participantID (PK/FK)

The ParticipantClassAttendance Table has 2 relationships and they are:

Table	Relationship	Other Table
ParticipantClassAttendance	Many-to-One	Participants
ParticipantClassAttendance	Many-to-One	ClassOffering

## OutOfHouse Table



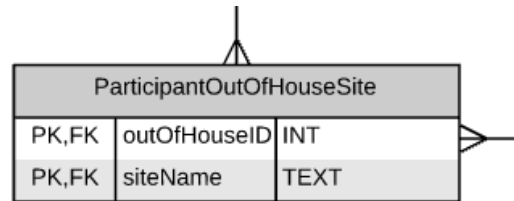
The OutOfHouse Table stores basic data about out-of-house sites descriptions. The table has the following attributes:

- outOfHouseID (PK/FK)
- description

The OutOfHouse Table has 2 relationships and they are:

Table	Relationship	Other Table
OutOfHouse	Zero or One-to-One	Participants
OutOfHouse	One-to-Many	ParticipantOutOfHouseSite

## ParticipantOutOfHouseSite Table



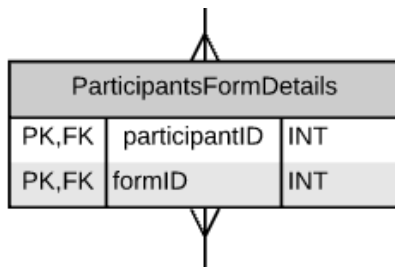
The ParticipantOutOfHouseSite Table stores basic data about out-of-house sites names. The table has the following attributes:

- outOfHouseID (PK/FK)
- siteName (PK/FK)

The ParticipantOutOfHouseSite Table has 2 relationships and they are:

Table	Relationship	Other Table
ParticipantOutOfHouseSite	Many-to-One	OutOfHouse
ParticipantOutOfHouseSite	Many-to-One	Sites

## ParticipantsFormDetails Table



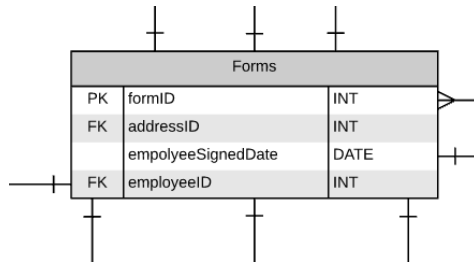
The ParticipantsFormDetails Table stores basic data connecting participants to forms. The table has the following attributes:

- participantID (PK/FK)
- formID (PK/FK)

The ParticipantsFormDetails Table has 2 relationships and they are:

Table	Relationship	Other Table
ParticipantsFormDetails	Many-to-One	Participants
ParticipantsFormDetails	Many-to-One	Forms

## Forms Table



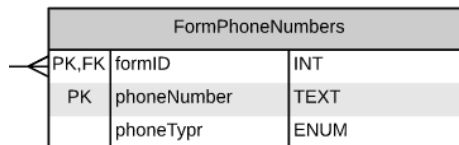
The Forms Table stores regarding forms. The table has the following attributes:

- formID (PK)
- addressID (FK)
- employeeSignedDate
- employeeID (FK)

The Forms Table has 9 relationships and they are:

Table	Relationship	Other Table
Forms	One-to-Many	ParticipantsFormDetails
Forms	One-to-Many	FormPhoneNumbers
Forms	Many-to-One	Employees
Forms	One-to-One	Addresses
Forms	One-to-Zero or One	SelfReferral
Forms	One-to-Zero or One	IntakeInformation
Forms	One-to-Zero or One	Surveys
Forms	One-to-Zero or One	AgencyReferral
Forms	One-to-Many	Family

## FormPhoneNumbers Table



The FormPhoneNumbers Table stores regarding forms. The table has the following attributes:

- formID (PK/FK)
- phoneNumber (PK)
- phoneType

The Forms Table has 1 relationship and it is:

Table	Relationship	Other Table
FormPhoneNumbers	Many-to-One	Forms

## Addresses Table

Addresses		
PK	addressID	INT
	addressNumber	INT
	street	TEXT
	apt	TEXT
FK	zipCode	INT

The Addresses Table stores basic data about addresses. The table has the following attributes:

- addressID (PK)
- addressNumber
- street
- apt
- zipCode (FK)

The Addresses Table has 2 relationships and they are:

Table	Relationship	Other Table
Addresses	Many-to-One	ZipCode
Addresses	One-to-One	Forms

## ZipCode Table

ZipCode		
PK	zipCode	INT
	city	TEXT
	state	TEXT

The ZipCode Table stores basic data about zip codes. The table has the following attributes:

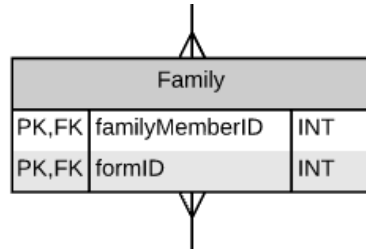
- zipCode (PK)
- city
- state

The ZipCode Table has 1 relationships and it is:

Table	Relationship	Other Table
ZipCode	One-to-Many	Addresses



## Family Table




The Family Table connects a generated ID to forms which mention family with additional data. The table has the following attributes:

- familyMemberID (PK/FK)
- formID (PK/FK)

The Family Table has 2 relationships and they are:

Table	Relationship	Other Table
Family	Many-to-One	Forms
Family	Many-to-One	FamilyMembers

## AgencyReferral Table



AgencyReferral		
PK,FK	agencyReferralID	INT
	reason	TEXT
	hasAgencyConsentForm	BOOLEAN
	additionalInfo	TEXT
	program	ENUM
	hasSpecialNeeds	BOOLEAN
	hasSubstanceAbuseHistory	BOOLEAN
	hasInvolvementCPS	BOOLEAN
	isPregnant	BOOLEAN
	hasIQDoc	BOOLEAN
	hasMentalHeath	BOOLEAN
	hasDomesticViolenceHistory	BOOLEAN
	childrenLiveWithIndividual	BOOLEAN
	dateFirstContact	DATE
	meansOfContact	TEXT
	dateOfInitialMeet	TIMESTAMP
	location	TEXT
	comments	TEXT


The AgencyReferral Table stores basic data from the referral form for a referral by an agency. The table has the following attributes:

- agencyReferralID (PK/FK)
- reason
- hasAgencyConsentForm
- additionalInfo
- program
- hasSpecialNeeds
- hasSubstanceAbuseHistory
- hasInvolvementCPS
- isPregnant
- hasIQDoc
- hasMentalHealth
- hasDomesticAbuseHistory
- childrenLiveWithIndividual
- dateFirstContact
- meansOfContact
- dateOfInitialMeet
- location
- comments

The AgencyReferral Table has 2 relationships and they are:

Table	Relationship	Other Table
AgencyReferral	Zero or One-to-One	Forms
AgencyReferral	One-to-Many	ContactAgencyAssociatedWithReferred

## Surveys Table



Surveys		
PK, FK	surveyID	INT
	materialPresentedScore	INT
	presTopicDiscussedScore	INT
	presOtherParentsScore	INT
	presChildPerspectiveScore	INT
	practiceInfoScore	INT
	recommendScore	INT
	suggestedFutureTopics	TEXT
	comments	TEXT


The Surveys Table stores data from the class surveys. The table has the following attributes:

- surveyID (PK)
- materialPresentedScore
- presTopicDiscussedScore
- presOtherParentsScore
- presChildPerspectiveScore
- practiceInfoScore
- recommendScore
- suggestedFutureTopics
- comments


The Surveys Table has 1 relationships and it is:

Table	Relationship	Other Table
Surveys	Zero or One-to-One	Forms

## IntakeInformation Table



IntakeInformation		
PK,FK	intakeInformationID	INT
	occupation	TEXT
	religion	TEXT
	ethnicity	TEXT
	handicapsOrMedication	TEXT
	lastYearOfSchoolCompleted	TEXT
	hasSubstanceAbuseHistory	BOOLEAN
	substanceAbuseDescription	TEXT
	timeSeparatedFromChildren	TEXT
	timeSeparatedFromPartner	TEXT
	relationshipToOtherParent	TEXT
	hasParentingPartnershipHistory	BOOLEAN
	hasInvolvementCPS	BOOLEAN
	previouslyInvolvedWithCPS	TEXT
	isMandatedToTakeClass	BOOLEAN
	mandatedByWhom	TEXT
	reasonForAttendance	TEXT
	safeParticipate	TEXT
	preventativeBehaviors	TEXT
	attendedOtherParentingClasses	BOOLEAN
	previousClassInfo	TEXT
	wasVictim	BOOLEAN



The IntakeInformation Table stores data from the intake forms. The table has the following attributes:

- intakeInformationID(PK/FK)
- occupation
- religion
- ethnicity
- handicapsOrMedication
- lastYearOfSchoolCompleted
- hasSubstanceAbuseHistory
- substanceAbuseDescription
- timeSeparatedFromChildren
- timeSeparatedFromPartner
- relationshipToOtherParent
- hasParentingPartnershipHistory
- hasInvolvementCPS
- previouslyInvolvedWithCPS
- isMandatedToTakeClass
- mandatedByWhom
- reasonForAttendance
- safeParticipate
- preventativeBehaviors
- attendedOtherParentingClasses
- previousClassInfo
- wasVictim

formOfChildhoodAbuse	TEXT
hasHadTherapy	BOOLEAN
feelStillHasIssuesFromChildAbuse	TEXT
mostImportantLikeToLearn	TEXT
hasDomesticViolenceHistory	BOOLEAN
hasDiscussedDomesticViolence	TEXT
hasHistoryOfViolenceInOriginFamily	BOOLEAN
hasHistoryOfViolenceInNuclearFamily	BOOLEAN
ordersOfProtectionInvolved	BOOLEAN
reasonForOrdersOfProtection	TEXT
hasBeenArrested	BOOLEAN
hasBeenConvicted	BOOLEAN
reasonForArrest/Conviction	TEXT
hasJailRecord	BOOLEAN
hasPrisonRecord	BOOLEAN
offenseForJailOrPrison	TEXT
currentlyOnParole	BOOLEAN
onParoleForWhatOffense	TEXT
prpFormSignedDate	DATE
ptpEnrollmentSignedDate	DATE
ptpConsentReleaseFormSignedDate	DATE

+

- formOfChildhoodAbuse
- hasHadTherapy
- feelStillHasIssuesFromChildAbuse
- mostImportantLikeToLearn
- hasDomesticViolenceHistory
- hasDiscussedDomesticViolence
- hasHistoryOfViolenceInOriginFamily
- hasHistoryOfViolenceInNuclearFamily
- ordersOfProtectionInvolved
- reasonForOrdersOfProtection
- hasBeenArrested
- hasBeenConvicted
- reasonForArrest/Conviction
- hasJailRecord
- hasPrisonRecord
- offenseForJailOrPrison
- currentlyOnParole
- onParoleForWhatOffense
- hasOtherFamilyMembersTakingClass
- ptpFormSignedDate
- ptpEnrollmentSignedDate
- ptpConsentReleaseFormSignedDate

The IntakeInformation Table has 3 relationships and they are:

Table	Relationship	Other Table
IntakeInformation	Zero or One-to-One	Forms
IntakeInformation	One-to-Many	ParticipantsIntakeLanguages
IntakeInformation	One-to-Many	EmergencyContactDetail

## ParticipantsIntakeLanguages Table



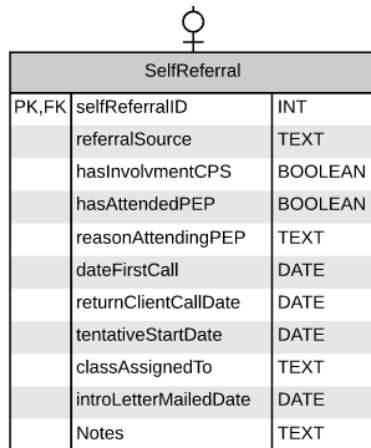
The ParticipantsIntakeLanguage Table stores basic data connecting intake forms to a language. The table has the following attributes:

- intakeInformationID (PK/FK)
- lang (PK/FK)

The ParticipantsIntakeLanguages Table has 2 relationships and they are:

Table	Relationship	Other Table
ParticipantsIntakeLanguage	Many-to-One	IntakeInformation
ParticipantsIntakeLanguage	Many-to-One	Languages

## SelfReferral Table



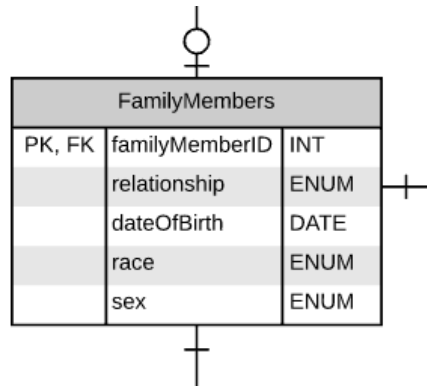
The SelfReferral Table stores data about the self-referral form. The table has the following attributes:

- selfReferralID (PK/FK)
- referralSource
- hasInvolvementCPS
- hasAttendedPEP
- dateFirstCall
- returnClientCallDate
- tentativeStartDate
- classAssignedTo
- introLetterMailedDate
- notes

The SelfReferral Table has 1 relationship and it is:

Table	Relationship	Other Table
SelfReferral	Zero or One-to-One	Forms

## FamilyMembers Table



The FamilyMembers Table stores basic data about family members.


The table has the following attributes:

- memberID (PK/FK)
- relationship
- dateOfBirth
- race
- sex

The FamilyMembers Table has 3 relationships and they are:

Table	Relationship	Other Table
FamilyMembers	Zero or One-to-One	People
FamilyMembers	One-to-Zero or One	Children
FamilyMembers	One-to-Many	Family

## Children Table



Children		
PK,FK	childrenID	INT
	custody	TEXT
	location	TEXT


The Children Table stores data regarding custody and location of children. The table has the following attributes:

- childrenID (PK/FK)
- custody
- location


The Children Table has 1 relationship and it is:

Table	Relationship	Other Table
Children	Zero or One-to-One	FamilyMembers

## ContactAgencyMembers Table



ContactAgencyMembers		
PK, FK	contactAgencyID	INT
	agency	ENUM
	phone	INT
	email	TEXT



The ContactAgencyMembers Table stores basic data about a contact agency. The table has the following attributes:

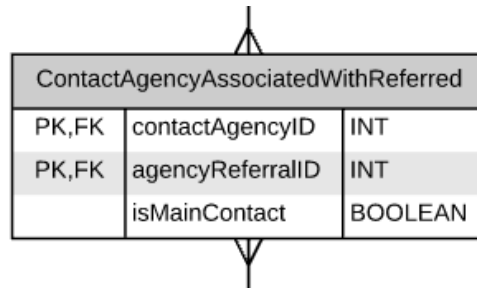
- contactAgencyID (PK/FK)
- agency
- phone
- email

The ContactAgencyMembers Table has 2 relationships and they are:

Table	Relationship	Other Table
ContactAgencyMembers	Zero or One-to-Many	People
ContactAgencyMembers	One-to-Many	ContactAgencyAssociatedWithReferred



## ContactAgencyAssociatedWithReferred Table



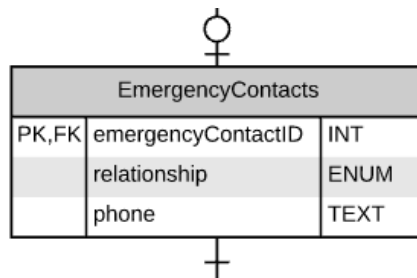
The ContactAgencyAssociatedWithReferred Table stores basic data connecting a contact agency with an agency referral. The table has the following attributes:

- contactAgencyID (PK/FK)
- agencyReferralID (PK/FK)
- isMainContact

The ContactAgencyAssociatedWithReferred Table has 2 relationships and they are:

Table	Relationship	Other Table
ContactAgencyAssociatedWithReferred	Many-to-One	ContactAgencyMembers
ContactAgencyAssociatedWithReferred	Many-to-One	AgencyReferral

## EmergencyContacts Table



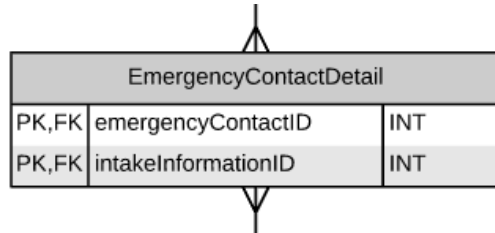
The EmergencyContacts Table stores basic data about emergency contacts. The table has the following attributes:

- emergencyContactID (PK/FK)
- relationship
- phone

The EmergencyContacts Table has 2 relationships and they are:

Table	Relationship	Other Table
EmergencyContacts	Zero or One-to-One	People
EmergencyContacts	One-to-Many	EmergencyContactDetail

## EmergencyContactDetail Table



The EmergencyContactDetail Table stores basic data about emergency contacts. The table has the following attributes:

- emergencyContactID (PK/FK)
- relationship
- phone

The EmergencyContactDetail Table has 2 relationships and they are:

Table	Relationship	Other Table
EmergencyContactDetail	Many-to-One	IntakeInformation
EmergencyContactDetail	Many-to-One	EmergencyContacts

## SQL Statements

### “DROP TABLE”

The “DROP TABLE IF EXISTS” statement in Structured Query Language (SQL) removes a table, if it exists with the provided name. The table would have previously been created with the “CREATE TABLE” statement. The dropped table is completely removed from the database schema and disk file. Once the table is removed, the file cannot be recovered—so use this statement with care.

```
DROP TABLE IF EXISTS
EmergencyContactDetail;

DROP TABLE IF EXISTS
ParticipantsIntakeLanguages;

DROP TABLE IF EXISTS Family;

DROP TABLE IF EXISTS
ContactAgencyAssociatedWithReferred
;

DROP TABLE IF EXISTS
IntakeInformation;

DROP TABLE IF EXISTS Surveys;

DROP TABLE IF EXISTS
AgencyReferral;

DROP TABLE IF EXISTS SelfReferral;

DROP TABLE IF EXISTS
ParticipantsFormDetails;

DROP TABLE IF EXISTS
FormPhoneNumbers;

DROP TABLE IF EXISTS Forms;

DROP TABLE IF EXISTS Addresses;

DROP TABLE IF EXISTS ZipCode;
```

```
DROP TABLE IF EXISTS
ParticipantOutOfHouseSite;

DROP TABLE IF EXISTS
FacilitatorLanguage;

DROP TABLE IF EXISTS
ParticipantClassAttendance;

DROP TABLE IF EXISTS
FacilitatorClassAttendance;

DROP TABLE IF EXISTS ClassOffering;

DROP TABLE IF EXISTS
CurriculumClasses;

DROP TABLE IF EXISTS Classes;

DROP TABLE IF EXISTS Curricula;

DROP TABLE IF EXISTS Sites;

DROP TABLE IF EXISTS Languages;

DROP TABLE IF EXISTS
ContactAgencyMembers;

DROP TABLE IF EXISTS
EmergencyContacts;

DROP TABLE IF EXISTS Children;

DROP TABLE IF EXISTS FamilyMembers;
```

```
DROP TABLE IF EXISTS OutOfHouse;

DROP TABLE IF EXISTS Participants;

DROP TABLE IF EXISTS Facilitators;

DROP TABLE IF EXISTS Employees;

DROP TABLE IF EXISTS People;

DROP TYPE IF EXISTS RELATIONSHIP;

DROP TYPE IF EXISTS PARENTINGPROGRAM;

DROP TYPE IF EXISTS PROGRAMTYPE;

DROP TYPE IF EXISTS PHONETYPE;

DROP TYPE IF EXISTS PERMISSION;

DROP TYPE IF EXISTS STATES;

DROP TYPE IF EXISTS LEVELTYPE;

DROP TYPE IF EXISTS CURRICULUMTYPE;

DROP TYPE IF EXISTS REFERRALTYPE;

DROP TYPE IF EXISTS FORM;

DROP TYPE IF EXISTS SEX;

DROP TYPE IF EXISTS RACE;
```

## “CREATE”

Create statements in SQL are used to create tables, types, and other objects that are a part of an ER Diagram.

### “CREATE TYPE AS ENUM”

The “CREATE TYPE AS ENUM” statement in SQL creates an enumerated data type with the name provided. Enumerated (enum) types are data types that comprise a static, ordered set of values.

```
CREATE TYPE RACE AS ENUM('Asian', 'Black', 'Latino', 'Native American', 'Pacific Islander', 'White');

CREATE TYPE SEX AS ENUM ('Male', 'Female');

CREATE TYPE FORM AS ENUM ('Intake', 'Referral');

CREATE TYPE REFERRALTYPE AS ENUM ('Self', 'Court', 'Agency', 'Friend', 'Family');

CREATE TYPE CURRICULUMTYPE AS ENUM ('FULL', 'MINI');

CREATE TYPE LEVELTYPE AS ENUM ('PRIMARY', 'SECONDARY');

CREATE TYPE STATES AS ENUM('Alabama', 'Alaska', 'Arizona', 'Arkansas', 'California', 'Colorado', 'Connecticut', 'Delaware',
'Florida', 'Georgia', 'Hawaii', 'Idaho', 'Illinois', 'Indiana', 'Iowa', 'Kansas', 'Kentucky', 'Louisiana', 'Maine', 'Maryland',
'Massachusetts', 'Michigan', 'Minnesota', 'Mississippi', 'Missouri', 'Montana', 'Nebraska', 'Nevada', 'New Hampshire',
'New Jersey', 'New Mexico', 'New York', 'North Carolina', 'North Dakota', 'Ohio', 'Oklahoma', 'Oregon', 'Pennsylvania',
'Rhode Island', 'South Carolina', 'South Dakota', 'Tennessee', 'Texas', 'Utah', 'Vermont', 'Virginia', 'Washington',
'West Virginia', 'Wisconsin', 'Wyoming');

CREATE TYPE PERMISSION AS ENUM('User', 'Facilitator', 'Administator', 'Superuser');

CREATE TYPE PHONETYPE AS ENUM('Primary', 'Secondary', 'Day', 'Evening', 'Home', 'Cell');

CREATE TYPE PROGRAMTYPE AS ENUM('In-House', 'Jail', 'Rehab');

CREATE TYPE PARENTINGPROGRAM AS ENUM('TPP', 'SNPP', 'PEP');

CREATE TYPE RELATIONSHIP AS ENUM ('Mother', 'Father', 'Daughter', 'Son', 'Sister', 'Brother', 'Aunt', 'Uncle', 'Niece',
'Nephew', 'Cousin', 'Grandmother', 'Grandfather', 'Granddaughter', 'Grandson', 'Stepsister', 'Stepbrother', 'Stepmother',
'Stepfather', 'Stepdaughter', 'Stepson', 'Sister-in-law', 'Brother-in-law', 'Mother-in-law', 'Daughter-in-law', 'Son-in-law',
'Friend', 'Other');
```

## “CREATE TABLE”

The “CREATE TABLE” statement in SQL defines a new table with the given parameters. The following “CREATE TABLE” statements are used to create the various tables in the database, corresponding with the tables in the ER Diagram.

### People and Subtypes

```
CREATE TABLE IF NOT EXISTS People (
    peopleID                SERIAL                NOT NULL    UNIQUE,
    firstName                TEXT                  NOT NULL,
    lastName                 TEXT                  NOT NULL,
    PRIMARY KEY (peopleID)
);

CREATE TABLE IF NOT EXISTS Employees (
    employeeID              INT,
    email                   TEXT                  NOT NULL,
    primaryPhone             TEXT,
    permissionLevel          PERMISSION          NOT NULL,
    PRIMARY KEY (employeeID),
    FOREIGN KEY (employeeID) REFERENCES People(peopleID)
);

CREATE TABLE IF NOT EXISTS Facilitators (
    facilitatorID            INT,
    program                  PARENTINGPROGRAM,
    PRIMARY KEY (facilitatorID),
    FOREIGN KEY (facilitatorID) REFERENCES Employees(employeeID)
);
```

```

CREATE TABLE IF NOT EXISTS Participants (
    participantID                INT,
    dateOfBirth                  DATE          NOT NULL,
    race                          RACE          NOT NULL,
    PRIMARY KEY (participantID),
    FOREIGN KEY (participantID) REFERENCES People(peopleID)
);

CREATE TABLE IF NOT EXISTS OutOfHouse (
    outOfHouseID                INT,
    description                   TEXT,
    PRIMARY KEY (outOfHouseID),
    FOREIGN KEY (outOfHouseID) REFERENCES Participants(participantID)
);

CREATE TABLE IF NOT EXISTS FamilyMembers (
    familyMemberID              INT,
    relationship                  RELATIONSHIP NOT NULL,
    dateOfBirth                  DATE          NOT NULL,
    race                          RACE,
    sex                           SEX,
    PRIMARY KEY (familyMemberID),
    FOREIGN KEY (familyMemberID) REFERENCES People(peopleID)
);

CREATE TABLE IF NOT EXISTS Children (
    childrenID                   INT,
    custody                       TEXT          NOT NULL,
    location                      TEXT          NOT NULL,
    PRIMARY KEY (childrenID),
    FOREIGN KEY (childrenID) REFERENCES FamilyMembers(familyMemberID)
);

```

```

CREATE TABLE IF NOT EXISTS EmergencyContacts (
    emergencyContactID          INT,
    relationship                 RELATIONSHIP      NOT NULL,
    primaryPhone                TEXT              NOT NULL,
    PRIMARY KEY (emergencyContactID),
    FOREIGN KEY (emergencyContactID) REFERENCES People(peopleID)
);

CREATE TABLE IF NOT EXISTS ContactAgencyMembers (
    contactAgencyID            INT,
    agency                     REFERRALTYPE      NOT NULL,
    phone                      TEXT              NOT NULL,
    email                     TEXT              NOT NULL,
    PRIMARY KEY (contactAgencyID),
    FOREIGN KEY (contactAgencyID) REFERENCES People(peopleID)
);

```

## Curricula and Class

```

CREATE TABLE IF NOT EXISTS Languages (
    lang                      TEXT              NOT NULL    UNIQUE,
    PRIMARY KEY (lang)
);

CREATE TABLE IF NOT EXISTS Sites (
    siteName                 TEXT              NOT NULL    UNIQUE,
    programType              PROGRAMTYPE      NOT NULL,
    PRIMARY KEY (siteName)
);

```

```

CREATE TABLE IF NOT EXISTS Curricula (
    curriculumID          SERIAL          NOT NULL      UNIQUE,
    curriculumName        TEXT           NOT NULL,
    curriculumType        PROGRAMTYPE    NOT NULL,
    missNumber            INT            DEFAULT 2,
    PRIMARY KEY (curriculumID)
);

CREATE TABLE IF NOT EXISTS Classes (
    topicName             TEXT           NOT NULL      UNIQUE,
    description            TEXT,
    PRIMARY KEY (topicName)
);

CREATE TABLE IF NOT EXISTS CurriculumClasses (
    topicName             TEXT,
    curriculumID          INT,
    PRIMARY KEY (topicName, curriculumID),
    FOREIGN KEY (topicName) REFERENCES Classes(topicName),
    FOREIGN KEY (curriculumID) REFERENCES Curricula(curriculumID)
);

CREATE TABLE IF NOT EXISTS ClassOffering (
    topicName             TEXT,
    date                  TIMESTAMP      NOT NULL,
    siteName              TEXT,
    lang                  TEXT          DEFAULT 'English',
    curriculumID          INT,
    PRIMARY KEY (topicName, date, siteName),
    FOREIGN KEY (topicName) REFERENCES Classes(topicName),
    FOREIGN KEY (siteName) REFERENCES Sites(siteName),
    FOREIGN KEY (lang) REFERENCES Languages(lang),
    FOREIGN KEY (curriculumID) REFERENCES Curricula(curriculumID)
);

```



```
CREATE TABLE IF NOT EXISTS FacilitatorClassAttendance (  
    topicName                TEXT,  
    date                     TIMESTAMP,  
    siteName                 TEXT,  
    facilitatorID            INT,  
    PRIMARY KEY (topicName, date, siteName, facilitatorID),  
    FOREIGN KEY (topicName, date, siteName) REFERENCES ClassOffering(topicName, date, siteName),  
    FOREIGN KEY (facilitatorID) REFERENCES Facilitators(facilitatorID)  
);  
  
CREATE TABLE IF NOT EXISTS ParticipantClassAttendance (  
    topicName                TEXT,  
    date                     TIMESTAMP,  
    siteName                 TEXT,  
    participantID            INT,  
    PRIMARY KEY (topicName, date, siteName, participantID),  
    FOREIGN KEY (topicName, date, siteName) REFERENCES ClassOffering(topicName, date, siteName),  
    FOREIGN KEY (participantID) REFERENCES Participants(participantID)  
);  
  
CREATE TABLE IF NOT EXISTS FacilitatorLanguage (  
    facilitatorID            INT,  
    lang                     TEXT,  
    level                     LEVELTYPE                NOT NULL,  
    PRIMARY KEY (facilitatorID, lang, level),  
    FOREIGN KEY (facilitatorID) REFERENCES Facilitators(facilitatorID),  
    FOREIGN KEY (lang) REFERENCES Languages(lang)  
);
```

```

CREATE TABLE IF NOT EXISTS ParticipantOutOfHouseSite (
    outOfHouseID INT,
    siteName TEXT,
    PRIMARY KEY (outOfHouseID, siteName),
    FOREIGN KEY (outOfHouseID) REFERENCES OutOfHouse(outOfHouseID),
    FOREIGN KEY (siteName) REFERENCES Sites(siteName)
);

```

### Forms and Related Tables

```

CREATE TABLE IF NOT EXISTS ZipCode (
    zipCode                INT                UNIQUE,
    city                   TEXT                NOT NULL,
    state                  STATES              NOT NULL,
    PRIMARY KEY (zipCode)
);

CREATE TABLE IF NOT EXISTS Addresses (
    addressID              SERIAL              NOT NULL    UNIQUE,
    addressNumber           INT,
    aptInfo                 TEXT,
    street                  TEXT                NOT NULL,
    zipCode                 INT                NOT NULL,
    PRIMARY KEY (addressID),
    FOREIGN KEY (zipCode) REFERENCES ZipCode(zipCode)
);

```

```

CREATE TABLE IF NOT EXISTS Forms (
    formID                SERIAL                NOT NULL    UNIQUE,
    addressID              INT,
    empolyeeSignedDate     DATE                NOT NULL    DEFAULT NOW(),
    employeeID             INT,
    PRIMARY KEY (formID),
    FOREIGN KEY (addressID) REFERENCES Addresses(addressID),
    FOREIGN KEY (employeeID) REFERENCES Employees(EmployeeID)
);

CREATE TABLE IF NOT EXISTS FormPhoneNumbers (
    formID INT,
    phoneNumber TEXT,
    phoneType PHONETYPE,
    PRIMARY KEY (formID, phoneNumber),
    FOREIGN KEY (formID) REFERENCES Forms(formID)
);

CREATE TABLE IF NOT EXISTS ParticipantsFormDetails (
    participantID          INT,
    formID                 INT,
    PRIMARY KEY (participantID, formID),
    FOREIGN KEY (participantID) REFERENCES Participants(participantID),
    FOREIGN KEY (formID) REFERENCES Forms(formID)
);

```

```

CREATE TABLE IF NOT EXISTS SelfReferral (
    selfReferralID                INT,
    referralSource                TEXT,
    hasInvolvementCPS             BOOLEAN,
    hasAttendedPEP                BOOLEAN,
    reasonAttendingPEP            TEXT,
    dateFirstCall                 DATE,
    returnClientCallDate          DATE,
    tentativeStartDate            DATE,
    classAssignedTo               TEXT,
    introLetterMailedDate         DATE,
    Notes                         TEXT,
    PRIMARY KEY (selfReferralID),
    FOREIGN KEY (selfReferralID) REFERENCES Forms(formID)
);

CREATE TABLE IF NOT EXISTS Surveys (
    surveyID                      INT,
    materialPresentedScore        INT,
    presTopicDiscussedScore       INT,
    presOtherParentsScore         INT,
    presChildPerspectiveScore     INT,
    practiceInfoScore             INT,
    recommendScore                INT,
    suggestedFutureTopics         TEXT,
    comments                      TEXT,
    PRIMARY KEY (surveyID),
    FOREIGN KEY (surveyID) REFERENCES Forms(formID)
);

```

```

CREATE TABLE IF NOT EXISTS AgencyReferral (
  agencyReferralID          INT,
  secondaryPhone             TEXT,
  reason                     TEXT,
  hasAgencyConstantForm    BOOLEAN,
  additionalInfo             TEXT,
  program                    PARENTINGPROGRAM,
  hasSpecialNeeds            BOOLEAN,
  hasSubstanceAbuseHistory  BOOLEAN,
  hasInvolvementCPS         BOOLEAN,
  isPregnant                 BOOLEAN,
  hasIQDoc                   BOOLEAN,
  hasMentalHeath             BOOLEAN,
  hasDomesticViolenceHistory BOOLEAN,
  childrenLiveWithIndividual BOOLEAN,
  dateFirstContact           DATE,
  meansOfContact             TEXT,
  dateOfInitialMeet          TIMESTAMP,
  location                   TEXT,
  comments                   TEXT,
  PRIMARY KEY (agencyReferralID),
  FOREIGN KEY (agencyReferralID) REFERENCES Forms(formID)
);

```

```

CREATE TABLE IF NOT EXISTS IntakeInformation (
  intakeInformationID          INT,
  secondaryPhone               TEXT,
  occupation                   TEXT,
  religion                     TEXT,
  ethnicity                    TEXT,
  handicapsOrMedication        TEXT,
  lastYearOfSchoolCompleted    TEXT,
  hasSubstanceAbuseHistory     BOOLEAN,
  substanceAbuseDescription     TEXT,
  timeSeparatedFromChildren    TEXT,
  timeSeparatedFromPartner     TEXT,
  relationshipToOtherParent    TEXT,
  hasParentingPartnershipHistory  BOOLEAN,
  hasInvolvementCPS            BOOLEAN,
  previouslyInvolvedWithCPS     TEXT,
  isMandatedToTakeClass        BOOLEAN,
  mandatedByWhom               TEXT,
  reasonForAttendance          TEXT,
  safeParticipate              TEXT,
  preventativeBehaviors        TEXT,
  attendedOtherParentingClasses  BOOLEAN,
  previousClassInfo             TEXT,
  wasVictim                    BOOLEAN,
  formOfChildhoodAbuse         TEXT,
  hasHadTherapy                BOOLEAN,

```

```

  feelStillHasIssuesFromChildAbuse  TEXT,
  mostImportantLikeToLearn           TEXT,
  hasDomesticViolenceHistory         BOOLEAN,
  hasDiscussedDomesticViolence       TEXT,
  hasHistoryOfViolenceInOriginFamily  BOOLEAN,
  hasHistoryOfViolenceInNuclearFamily  BOOLEAN,
  ordersOfProtectionInvolved          BOOLEAN,
  reasonForOrdersOfProtection         TEXT,
  hasBeenArrested                    BOOLEAN,
  hasBeenConvicted                   BOOLEAN,
  reasonForArrestOrConviction         TEXT,
  hasJailRecord                      BOOLEAN,
  hasPrisonRecord                    BOOLEAN,
  offenseForJailOrPrison              TEXT,
  currentlyOnParole                  BOOLEAN,
  onParoleForWhatOffense              TEXT,
  prpFormSignedDate                  DATE,
  ptpEnrollmentSignedDate            DATE,
  ptpConstatReleaseFormSignedDate    DATE,
  PRIMARY KEY (intakeInformationID),
  FOREIGN KEY (intakeInformationID) REFERENCES
Forms(formID)
);

```

```
CREATE TABLE IF NOT EXISTS ContactAgencyAssociatedWithReferred (
    contactAgencyID INT,
    agencyReferralID INT,
    isMainContact BOOLEAN,
    PRIMARY KEY (contactAgencyID, agencyReferralID),
    FOREIGN KEY (contactAgencyID) REFERENCES ContactAgencyMembers(contactAgencyID),
    FOREIGN KEY (agencyReferralID) REFERENCES AgencyReferral(agencyReferralID)
);

CREATE TABLE IF NOT EXISTS Family (
    familyMembersID INT,
    formID INT,
    PRIMARY KEY (familyMembersID, formID),
    FOREIGN KEY (familyMembersID) REFERENCES FamilyMembers(familyMemberID),
    FOREIGN KEY (formID) REFERENCES Forms(formID)
);

CREATE TABLE IF NOT EXISTS ParticipantsIntakeLanguages (
    intakeInformationID INT,
    lang TEXT,
    PRIMARY KEY (intakeInformationID, lang),
    FOREIGN KEY (intakeInformationID) REFERENCES IntakeInformation(intakeInformationID),
    FOREIGN KEY (lang) REFERENCES Languages(lang)
);

CREATE TABLE IF NOT EXISTS EmergencyContactDetail (
    emergencyContactID INT,
    intakeInformationID INT,
    PRIMARY KEY (emergencyContactID, intakeInformationID),
    FOREIGN KEY (emergencyContactID) REFERENCES EmergencyContacts(emergencyContactID),
    FOREIGN KEY (intakeInformationID) REFERENCES IntakeInformation(intakeInformationID)
);
```