**Amy McVicar**

**659 Data Administration Concepts and Database Management**

**Fall 2019 - Thursday 9 PM EDT**

**Project Deliverable**

**Dog Project**

Contents

[Part 1: Design 2](#_Toc27415741)

[Summary 2](#_Toc27415742)

[Business Case 2](#_Toc27415743)

[Stakeholders 2](#_Toc27415744)

[High-Level Rules 3](#_Toc27415745)

[Expectations 3](#_Toc27415746)

[Source Data Samples 3](#_Toc27415747)

[Entity Relationship Diagram 4](#_Toc27415748)

[Data Dictionary 4](#_Toc27415749)

[Part 2: Implementation 10](#_Toc27415750)

[Logical Model 10](#_Toc27415751)

[SQL DDL 11](#_Toc27415752)

[SQL DML 19](#_Toc27415753)

[Answers to Data Questions 20](#_Toc27415754)

[GUI Prototype 22](#_Toc27415755)

[Reflection 25](#_Toc27415756)



# **Part 1: Design**

## **Summary**

The following document describes a database created to support any organization focused on helping canines in need. Organizations do not need to be nonprofit and can start with a for-profit model to assist getting off the ground with fewer requirements. Organizations may also help and interact with other organizations in a multitude of ways. This document outlines a database designed to assist with a variety of scenarios an organization supporting canine rescue and assistance may find itself.

## **Business Case**

There are many types of organizations that strive to assist canines in need. They can range from profit to non-profit, to those that directly support or those that support other organizations indirectly. They all need data management and guidance. This document will outline a database for any scenario and will be offered for free.

The data and database should answer five data questions to ensure successful adoption by and support for an organization.

1. **What dogs are in the database, and what are their statuses?**
2. **How much money was taken in a period?**
3. **How much money was spent in a period?**
4. **What are the details and sources of money?**
5. **What are the details and sources of the costs?**

## **Stakeholders**

The high-level stakeholders are people, organizations, and (most importantly) canines.

* + - Canines
    - People
      * Donors
      * Sponsors
      * Customers
      * Vendors
      * Partners
      * Employees
      * Volunteers
      * Adopters
    - Organizations
      * Rescues
      * Shelters
      * Vendors
      * Partners

## **High-Level Rules**

**People & Organizations**

* A contact or an organization can own a dog.
* Contacts and organizations can be associated with multiple dogs.

**Dogs**

* Dogs are not one-time clients but can have multiple status changes.
* A dog can be owned by contact or by an organization.
* A dog can be directly associated with costs or income (in the case it is sponsored, for example).

**Money**

* Costs can be associated with one dog, but if associated with more than one dog, it will not be assigned to dogs.
* Income can be a donation that would have nothing due, a sponsorship which would only have estimated due within cancellation policy and contractually obligated amounts owing.

**Events**

* Events may have registrants or may be non-registered events.
* Events may be online, offline or other fund-raising efforts

**Products**

* The product table is a simple inventory tool. A future phase may add functionality.

## **Expectations**

The result will be a data model that can assist data capture, management, and web support for any organization that needs to start collecting data and helping dogs regardless of the business model.

## **Source Data Samples**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Current Name** | **Original Name** | **Breed** | **Color** | **Sex** | **Age Estimated** | **Birthdate** | **Status** |
| Dixie | Unknown | StumpyX | Orange | Female | Y | Jun-13 | Adopted |
| Cody | Cody | StumpyX | Merle | Male | Y | Nov-14 | Adopted |
| Fennec | Unknown | ACD Mis | Beige | Male | Y | Jan-19 | Foster |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **First Name** | **Last Name** | **Email** | **Phone** | **Type** |
| Gary | Schoonover | [doofusdogs@nodomain.xyz](mailto:doofusdogs@nodomain.xyz) | 415-497-6382 | Volunteer |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Product** | **Size** | **Description** | **Color** | **Cost** | **List Price** |
| Dog Jacket | 30-40 lbs | Jacket for 30-40 lbs Dog | Multi | 10 | 15 |
| Canvas Shopping Bag | Medium | Canvas Shopping Bag | Cream | 5 | 30 |

## **Entity Relationship Diagram**

A screenshot of a cell phone

Description automatically generated

## **Data Dictionary**

| **Entity** | **Attribute** | **Type** | **Properties** |
| --- | --- | --- | --- |
| Organizations | Org\_ID | int identity | PK |
| Organizations | Org\_Name | varchar(100) | Required, Unique |
| Organizations | Org\_Address\_1 | varchar(100) |  |
| Organizations | Org\_Address\_2 | varchar(100) |  |
| Organizations | Org\_City | varchar(100) |  |
| Organizations | Org\_State | varchar(100) | State or Provence |
| Organizations | Org\_Postal | varchar(50) |  |
| Organizations | Org\_Country\_ID | int | Required, FK |
| Organizations | Org\_Phone | varchar(50) |  |
| Organizations | Org\_URL | varchar(100) |  |
| Organizations | Contact\_Org\_Status\_ID | char(20) | Required, Active or Inactive (Unknown is Active) |
| Organizations | Org\_Type\_ID | int | FK |
| Organization\_Type | Org\_Type\_ID | int identity | PK |
| Organization\_Type | Org\_Type\_Value | char(20) | Required, Unique Values: Vendor Rescue Shelter Profit Partner |
| Countries | Country\_ID | int identity | PK |
| Countries | Country\_ISO\_Code | char(2) | 2 char iso country code standard |
| Countries | Country\_Name | char(200) | country name (iso code standard) |
| Contacts | Contact\_ID | int identity | PK |
| Contacts | First\_Name | varchar(50) |  |
| Contacts | Middle\_Name | varchar(50) |  |
| Contacts | Last\_Name | varchar(50) |  |
| Contacts | Contact\_Org\_ID | int | Required, FK, defines if contact is personal or linked to an organization |
| Contacts | Phone | varchar(50) |  |
| Contacts | Email | varchar(100) | Required, Unique |
| Contacts | Email\_Opt\_In | char(1) | Y, N, Null |
| Contacts | Address\_1 | varchar(100) |  |
| Contacts | Address\_2 | varchar(100) |  |
| Contacts | City | varchar(100) |  |
| Contacts | State | varchar(100) |  |
| Contacts | Postal | varchar(50) |  |
| Contacts | Country\_ID | int | Required, FK |
| Contacts | Contact\_Org\_Status\_ID | int | Required, Default '1' for Active |
| Contacts | Org\_ID | int | FK |
| Contact\_Organization | Contact\_Org\_ID | int identity | PK |
| Contact\_Organization | Contact\_Org\_Value | char(50) | Values:  Organization Personal |
| Contact\_Sub\_Type\_Values | Contact\_Sub\_Type\_Value\_ID | int identity | PK |
| Contact\_Sub\_Type\_Values | Contact\_Sub\_Type\_Value | char(50) | Values:  Organization - Partner Organization - Vendor Personal - Donor Personal - Employee Personal - Volunteer Personal - Foster Personal - Adopter Personal - Other |
| Contact\_Sub\_Type | Contact\_Sub\_Type\_ID | int identity | PK |
| Contact\_Sub\_Type | Contact\_ID | int | FK |
| Contact\_Sub\_Type | Contact\_Sub\_Type\_Value\_ID | int | FK |
| Contact\_Sub\_Type | Contact\_Type\_Date | datetime | Date entered. Default to current, |
| Contact\_Org\_Status | Contact\_Org\_Status\_ID | int identity | PK |
| Contact\_Org\_Status | Contact\_Org\_Status\_Value | char(50) | Values: Active Inactive - Deceased Inactive - Out of Business Inactive - Other Inactive - Bad Contact Data |
| Canines | Canine\_ID | int identity | PK |
| Canines | Current\_Name | varchar(50) | Current Name |
| Canines | Original\_Name | varchar(50) | If known previous name |
| Canines | Age\_Estimated | char(1) | Y/N |
| Canines | Birthdate | date | YYYY-MM-DD |
| Canines | Breed | varchar(50) | Text Field, usually the best guess |
| Canines | Description | varchar(255) |  |
| Canines | Canine\_Location\_Type\_ID | int | Required, FK |
| Canines | Canine\_Status\_ID | in | Required, FK |
| Canines | Contact\_Org\_ID | int | Required, FK |
| Canines | Owner\_Org\_ID | int | FK |
| Canines | Owner\_Contact\_ID | int | FK |
| Canines | Owner\_Org\_Identifier | char(100) | If Owner Organization has ID# for dog |
| Canines | Create\_Date | DateTime | Required, Default to current |
| Canine\_Location\_Type | Canine\_Location\_Type\_ID | int identity | PK |
| Canine\_Location\_Type | Canine\_Location\_Type\_Value | char(20) | Values: Shelter Fostered Foster Hospice Home Rescue |
| Canine\_Status | Canine\_Status\_ID | ind identity | PK |
| Canine\_Status | Canine\_Status\_Value | char(20) | Values: Urgent Need Need Fostered Sponsored Adopted Lost Found Reunited Deceased |
| Income | Income\_ID | int identity | PK |
| Income | Contact\_Org\_ID | varchar(50) | FK, Defines if the record is associated thru a contact record or directly to an organization |
| Income | Org\_ID | varchar(50) | FK |
| Income | Contact\_ID | varchar(50) | FK |
| Income | Amount\_Due | decimal (10,2) |  |
| Income | Amount\_Paid | decimal (10,2) | Required |
| Income | Due\_Date | date |  |
| Income | Paid\_Date | date |  |
| Income | Payment\_Type\_ID | int | FK |
| Income | Payment\_Description | varchar(255) |  |
| Income | Event\_ID | int | FK |
| Income | Canine\_ID | int | FK |
| Payment\_Type | Payment\_Type\_ID | int identity | PK |
| Payment\_Type | Payment\_Type\_Value | char(20) | Values: Adoption Fee Sponsorship Donation Customer Credit Adjustment Other |
| Costs | Cost\_ID | int identity | PK |
| Costs | Contact\_Org\_ID | int | FK |
| Costs | Org\_ID | int | FK |
| Costs | Contact\_ID | int | FK |
| Costs | Cost\_Type\_ID | int | FK |
| Costs | Cost\_Description | varchar(255) |  |
| Costs | PO | varchar(50) |  |
| Costs | Amount\_Due | decimal (10,2) | Required |
| Costs | Amount\_Paid | decimal (10,2) |  |
| Costs | Due\_Date | date | YYYY-MM-DD |
| Costs | Paid\_Date | date | YYYY-MM-DD |
| Costs | Paid\_By\_Contact\_ID | int | FK |
| Costs | Event\_ID | int | FK |
| Costs | Canine\_ID | int | FK |
| Cost\_Type | Cost\_Type\_ID | int identity | PK |
| Cost\_Type | Cost\_Type\_Value | char(50) | Values (vary by org): Payroll Supplies Veterinarian Taxes Services Other |
| Products | Products\_ID | int identity | PK |
| Products | Product\_Name | varchar(100) |  |
| Products | Product\_Type\_ID | int | FK |
| Products | Product\_Description | varchar(255) |  |
| Products | Product\_Cost | decimal (10,2) |  |
| Products | Sale\_Price | decimal (10,2) |  |
| Products | Inventory\_Quantity | int |  |
| Products | Product\_Graphic\_ID | int |  |
| Products | Product\_Status | char(20) | Required: Active or Inactive (Default to Active) |
| Product\_Type | Product\_Type\_ID | int identity |  |
| Product\_Type | Product\_Type\_Value | char(100) | Unique, Required. Values vary by Org. Values: Collar Leash Food Mat Dog Bed Padding Dog Bed Cover Travel Water Bowl Hiking Water Dispenser |
| Events | Event\_ID | int identity | PK |
| Events | Event\_Name | varchar(100) | Required |
| Events | Location | varchar(50) | Required |
| Events | Event\_Start\_Date | datetime | Required |
| Events | Event\_End\_Date | datetime |  |
| Events | Event\_Type\_ID | int |  |
| Events | Description | varchar(255) |  |
| Event\_Type | Event\_Type\_ID | int identity |  |
| Event\_Type | Event\_Type\_Value | char(100) | Unique, Required. Values vary by Org. Values: Adoption Auction Awareness Donation Drive Sales Other |



# **Part 2: Implementation**

## **Logical Model**

A close up of text on a white background

Description automatically generated

## SQL DDL

/\*

SQL DDL Statements for Dog Project

Author: Amy McVicar

December 2019

IST659 Database Project

\*/



/\*

Drop Tables

Create Tables

\*/

DROP TABLE IF EXISTS Income;

DROP TABLE IF EXISTS Costs;

DROP TABLE IF EXISTS Canines;

DROP TABLE IF EXISTS Contacts;

DROP TABLE IF EXISTS Organizations;

DROP TABLE IF EXISTS Fun\_Events;

DROP TABLE IF EXISTS Products;

DROP TABLE IF EXISTS Contact\_Organization;

DROP TABLE IF EXISTS Organization\_Type;

DROP TABLE IF EXISTS Event\_Type;

DROP TABLE IF EXISTS Countries;

DROP TABLE IF EXISTS Contact\_Sub\_Type\_Value;

DROP TABLE IF EXISTS Contact\_Sub\_Type;

DROP TABLE IF EXISTS Contact\_Org\_Status;

DROP TABLE IF EXISTS Canine\_Location\_Type;

DROP TABLE IF EXISTS Canine\_Status;

DROP TABLE IF EXISTS Cost\_Type;

DROP TABLE IF EXISTS Payment\_Type;

DROP TABLE IF EXISTS Product\_Type;

---------------------------------------------

---------------------------------------------

-- Create Table Organization\_Type

CREATE TABLE Organization\_Type (

-- Columns for the Organization\_Type table

Org\_Type\_ID int identity,

Org\_Type\_Value char(20) not null,

-- Constraints on the Organization\_Type Table

CONSTRAINT PK\_Organization\_Type PRIMARY KEY (Org\_Type\_ID),

CONSTRAINT U1\_Organization\_Type UNIQUE (Org\_Type\_Value)

)

-- End Creating the Organization\_Type table

---------------------------------------------

-- Create Table Event\_Type

CREATE TABLE Event\_Type (

-- Columns for the Event\_Type table

Event\_Type\_ID int identity,

Event\_Type\_Value char(100) NOT NULL,

-- Constraints on the Event\_Type Table

CONSTRAINT PK\_Event\_Type PRIMARY KEY (Event\_Type\_ID),

CONSTRAINT U1\_Event\_Type UNIQUE (Event\_Type\_Value)

)

-- End Creating the Event\_Type table

---------------------------------------------

-- Create Table Countries

CREATE TABLE Countries (

-- Columns for the Countries table

Country\_ID int identity,

Country\_ISO\_Code char(2) not null,

Country\_Name char(200) not null,

-- Constraints on the Countries Table

CONSTRAINT PK\_Countries PRIMARY KEY (Country\_ID),

CONSTRAINT U1\_Countries UNIQUE (Country\_ISO\_Code),

CONSTRAINT U2\_Countries UNIQUE (Country\_Name)

)

-- End Creating the Countries table

---------------------------------------------

-- Create Table Contact\_Sub\_Type\_Value

CREATE TABLE Contact\_Sub\_Type\_Value (

-- Columns for the Contact\_Sub\_Type\_Value table

Contact\_Sub\_Type\_Value\_ID int identity,

Contact\_Sub\_Type\_Value char(50),

-- Constraints on the Contact\_Sub\_Type\_Valuee Table

CONSTRAINT PK\_Contact\_Sub\_Type\_Value PRIMARY KEY (Contact\_Sub\_Type\_Value\_ID),

CONSTRAINT U1\_Contact\_Sub\_Type\_Value UNIQUE (Contact\_Sub\_Type\_Value)

)

-- End Creating the Contact\_Sub\_Type\_Value table

---------------------------------------------

-- Create Table Contact\_Sub\_Type

CREATE TABLE Contact\_Sub\_Type (

-- Columns for Contact\_Sub\_Type table

Contact\_Sub\_Type\_ID int identity,

Contact\_ID int NOT NULL,

Contact\_Sub\_Typ\_Value\_ID int NOT NULL,

Contact\_Type\_Date datetime DEFAULT CURRENT\_TIMESTAMP,

-- Constraints on the Contact\_Sub\_Type Table

CONSTRAINT PK\_Contact\_Sub\_Type PRIMARY KEY (Contact\_Sub\_Type\_ID),

)

-- End Creating the Contact\_Sub\_Type table

---------------------------------------------

-- Create Table Contact\_Org\_Status

CREATE TABLE Contact\_Org\_Status (

-- Columns for the Contact\_Status table

Contact\_Org\_Status\_ID int identity,

Contact\_Org\_Status\_Value char(50) NOT NULL,

-- Constraints on the Organization\_Type Table

CONSTRAINT PK\_CContact\_Org\_Status PRIMARY KEY (Contact\_Org\_Status\_ID),

CONSTRAINT U1\_Contact\_Org\_Status UNIQUE (Contact\_Org\_Status\_Value)

)

-- End Creating the Contact\_Status table

---------------------------------------------

-- Create Table Canine\_Location\_Type

CREATE TABLE Canine\_Location\_Type (

-- Columns for the Canine\_Location\_Type table

Canine\_Location\_Type\_ID int identity,

Canine\_Location\_Type\_Value char(20) NOT NULL,

-- Constraints on the Canine\_Location\_Type Table

CONSTRAINT PK\_Canine\_Location\_Type PRIMARY KEY (Canine\_Location\_Type\_ID),

CONSTRAINT U1\_Canine\_Location\_Type UNIQUE (Canine\_Location\_Type\_Value)

)

-- End Creating the Canine\_Location\_Type table

---------------------------------------------

-- Create Table Canine\_Status

CREATE TABLE Canine\_Status (

-- Columns for the Canine\_Status table

Canine\_Status\_ID int identity,

Canine\_Status\_Value char(20) NOT NULL,

-- Constraints on the Canine\_Status Table

CONSTRAINT PK\_Canine\_Status PRIMARY KEY (Canine\_Status\_ID),

CONSTRAINT U1\_Canine\_Status UNIQUE (Canine\_Status\_Value)

)

-- End Creating the Canine\_Status table

---------------------------------------------

-- Create Table Payment\_Type

CREATE TABLE Payment\_Type (

-- Columns for the Payment\_Type table

Payment\_Type\_ID int identity,

Payment\_Type\_Value char(20) NOT NULL,

-- Constraints on the Payment\_Type Table

CONSTRAINT PK\_Payment\_Type PRIMARY KEY (Payment\_Type\_ID),

CONSTRAINT U1\_Payment\_Type UNIQUE (Payment\_Type\_Value)

)

-- End Creating the Payment\_Type table

---------------------------------------------

-- Create Table Cost\_Type

CREATE TABLE Cost\_Type (

-- Columns for the Cost\_Type table

Cost\_Type\_ID int identity,

Cost\_Type\_Value char(20) NOT NULL,

-- Constraints on the Cost\_Type Table

CONSTRAINT PK\_Cost\_Type PRIMARY KEY (Cost\_Type\_ID),

CONSTRAINT U1\_Cost\_Type UNIQUE (Cost\_Type\_Value)

)

-- End Creating the Cost\_Type table

---------------------------------------------

-- Create Table Product\_Type

CREATE TABLE Product\_Type (

-- Columns for the Product\_Type table

Product\_Type\_ID int identity,

Product\_Type\_Value char(100) NOT NULL,

-- Constraints on the Product\_Type Table

CONSTRAINT PK\_Product\_Type PRIMARY KEY (Product\_Type\_ID),

CONSTRAINT U1\_Product\_Type UNIQUE (Product\_Type\_Value)

)

-- End Creating the Product\_Type table

-- Create Table Contact\_Organization

CREATE TABLE Contact\_Organization (

-- Columns for the Contact\_Organization table

Contact\_Org\_ID int identity,

Contact\_Org\_Value char(50) not null,

-- Constraints on the Contact\_Organization Table

CONSTRAINT PK\_Contact\_Organization PRIMARY KEY (Contact\_Org\_ID),

CONSTRAINT U1\_Contact\_Organization UNIQUE (Contact\_Org\_Value)

)

-- End Creating the Contact\_Organization table

---------------------------------------------

-- Create Fun\_Events Table

CREATE TABLE Fun\_Events (

-- Columns for the Fun\_Events table

Event\_ID int identity,

Event\_Name varchar(100) NOT NULL,

Location varchar(50) NOT NULL,

Event\_Start\_Date date NOT NUll,

Event\_End\_Date date,

Event\_Type\_ID int,

Description varchar(255),

-- Constraints on the Fun\_Events Table

CONSTRAINT PK\_Fun\_Events PRIMARY KEY (Event\_ID),

CONSTRAINT F1\_Fun\_Events FOREIGN KEY (Event\_Type\_ID) REFERENCES Event\_Type(Event\_Type\_ID)

)

-- End Creating the Fun\_Events table

---------------------------------------------

-- Create Products Table

CREATE TABLE Products (

-- Columns for the Products table

Product\_ID int identity,

Product\_Name varchar(100),

Product\_Type int,

Product\_Description varchar(255),

Product\_Cost decimal(10,2),

Sale\_Price decimal(10,2),

Inventory\_Quantity int,

Product\_Graphic\_ID int,

Product\_Status char(20) DEFAULT 'Active',

-- Constraints on the Products Table

CONSTRAINT PK\_Products PRIMARY KEY (Product\_ID)

)

-- End Creating the Products table

-- Create Table Organizations

CREATE TABLE Organizations (

-- Columns for the User table

Org\_ID int identity,

Org\_Name varchar(100) not null,

Org\_Address\_1 varchar(100),

Org\_Address\_2 varchar(100),

Org\_City varchar(100),

Org\_State varchar(100),

Org\_Postal varchar(50),

Org\_Country\_ID int,

Org\_Phone varchar(50),

Org\_URL varchar(100),

Contact\_Org\_Status\_ID int DEFAULT 1,

Org\_Type\_ID int,

-- Constraints on the Organizations Table

CONSTRAINT PK\_Organizations PRIMARY KEY (Org\_ID),

CONSTRAINT U1\_Organizations UNIQUE (Org\_Name),

CONSTRAINT FK1\_Organizations FOREIGN KEY (Org\_Type\_ID) REFERENCES Organization\_Type(Org\_Type\_ID),

CONSTRAINT FK2\_Organizations FOREIGN KEY (Contact\_Org\_Status\_ID) REFERENCES Contact\_Org\_Status(Contact\_Org\_Status\_ID)

)

-- End Creating the Organizations table

--------------------------------------------

---------------------------------------------

-- Create Table Contacts

CREATE TABLE Contacts (

-- Columns for the Contacts table

Contact\_ID int identity,

First\_Name varchar(50),

Middle\_Name varchar(50),

Last\_Name varchar(50),

Contact\_Org\_ID int,

Phone varchar(50),

Email varchar(100) not null,

Email\_Opt\_In char(1),

Address\_1 varchar(100),

Address\_2 varchar(100),

City varchar(100),

Postal varchar(50),

[State] varchar(100),

Country\_ID int,

Contact\_Org\_Status\_ID int DEFAULT 1,

Org\_ID int,

-- Constraints on the Contacts Table

CONSTRAINT PK\_Contacts PRIMARY KEY (Contact\_ID),

CONSTRAINT U1\_Contacts UNIQUE (Email),

CONSTRAINT FK1\_Contacts FOREIGN KEY (Contact\_Org\_ID) REFERENCES Contact\_Organization(Contact\_Org\_ID),

CONSTRAINT FK2\_Contacts FOREIGN KEY (Country\_ID) REFERENCES Countries(Country\_ID),

CONSTRAINT FK3\_Contacts FOREIGN KEY (Org\_ID) REFERENCES Organizations(Org\_ID),

CONSTRAINT FK4\_Contacts FOREIGN KEY (Contact\_Org\_Status\_ID) REFERENCES Contact\_Org\_Status(Contact\_Org\_Status\_ID)

)

-- End Creating the Contacts table

---------------------------------------------

-- Create Table Canines

CREATE TABLE Canines (

-- Columns for the Canines table

Canine\_ID int identity,

Current\_Name varchar(50) NOT NULL,

Original\_Name varchar(50),

Age\_Estimated char(1),

Birthdate date,

Breed varchar(50),

Description varchar(255),

Canine\_Location\_Type\_ID int NOT NULL,

Canine\_Status\_ID int NOT NULL,

Contact\_Org\_ID int NOT NULL,

Owner\_Org\_ID int,

Owner\_Contact\_ID int,

Owner\_Organization\_Identifier char(100),

Create\_Date datetime,

-- Constraints on the Canines Table

CONSTRAINT PK\_Canines PRIMARY KEY (Canine\_ID),

CONSTRAINT F1\_Canines FOREIGN KEY (Canine\_Location\_Type\_ID) REFERENCES Canine\_Location\_Type(Canine\_Location\_Type\_ID),

CONSTRAINT F2\_Canines FOREIGN KEY (Canine\_Status\_ID) REFERENCES Canine\_Status(Canine\_Status\_ID),

CONSTRAINT F3\_Canines FOREIGN KEY (Contact\_Org\_ID) REFERENCES Contact\_Organization(Contact\_Org\_ID),

CONSTRAINT F4\_Canines FOREIGN KEY (Owner\_Org\_ID) REFERENCES Organizations(Org\_ID),

CONSTRAINT F5\_Canines FOREIGN KEY (Owner\_Contact\_ID) REFERENCES Contacts(Contact\_ID)

)

-- End Creating the Canines table

---------------------------------------------

---------------------------------------------

-- Create Table Income

CREATE TABLE Income (

-- Columns for the Income table

Income\_ID int identity,

Contact\_Org\_ID int,

Org\_ID int,

Contact\_ID int,

Amount\_Due decimal(10,2),

Amount\_Paid decimal(10,2) NOT NULL,

Due\_Date date,

Paid\_Date date NOT NULL,

Payment\_Type\_ID int,

Payment\_Description varchar(255),

Event\_ID int,

Canine\_ID int,

-- Constraints on the Income table

CONSTRAINT PK\_Income PRIMARY KEY (Income\_ID),

CONSTRAINT F1\_Income FOREIGN KEY (Contact\_Org\_ID) REFERENCES Contact\_Organization(Contact\_Org\_ID),

CONSTRAINT F2\_Income FOREIGN KEY (Org\_ID) REFERENCES Organizations(Org\_ID),

CONSTRAINT F3\_Income FOREIGN KEY (Contact\_ID) REFERENCES Contacts(Contact\_ID),

CONSTRAINT F4\_Income FOREIGN KEY (Payment\_Type\_ID) REFERENCES Payment\_Type(Payment\_Type\_ID),

CONSTRAINT F5\_Income FOREIGN KEY (Event\_ID) REFERENCES Fun\_Events(Event\_ID),

CONSTRAINT F6\_Income FOREIGN KEY (Canine\_ID) REFERENCES Canines(Canine\_ID)

)

-- End Creating the Income table

---------------------------------------------

-- Create Table Costs

CREATE TABLE Costs (

-- Columns for the Costs table

Cost\_ID int identity,

Contact\_Org\_ID int,

Org\_ID int,

Contact\_ID int,

Cost\_Type\_ID int,

Cost\_Description varchar(255),

PO varchar(50),

Amount\_Due decimal(10,2) NOT NULL,

Due\_Date date NOT NULL,

Paid\_Date date,

Paid\_By\_Contact\_ID int,

Event\_ID int,

Canine\_ID int

-- Constraints on the Costs Table

CONSTRAINT PK\_Costs PRIMARY KEY (Cost\_ID),

CONSTRAINT F1\_Costs FOREIGN KEY (Contact\_Org\_ID) REFERENCES Contact\_Organization(Contact\_Org\_ID),

CONSTRAINT F2\_Costs FOREIGN KEY (Org\_ID) REFERENCES Organizations(Org\_ID),

CONSTRAINT F3\_Costs FOREIGN KEY (Contact\_ID) REFERENCES Contacts(Contact\_ID),

CONSTRAINT F4\_Costs FOREIGN KEY (Cost\_Type\_ID) REFERENCES Cost\_Type(Cost\_Type\_ID),

CONSTRAINT F5\_Costs FOREIGN KEY (Paid\_By\_Contact\_ID) REFERENCES Contacts(Contact\_ID),

CONSTRAINT F6\_Costs FOREIGN KEY (Event\_ID) REFERENCES [Fun\_Events](Event\_ID),

CONSTRAINT F7\_Costs FOREIGN KEY (Canine\_ID) REFERENCES Canines(Canine\_ID)

)

-- End Creating the Costs table

## SQL DML

/\*

SQL DML Statements for Dog Project

Author: Amy McVicar

December 2019

IST659 Database Project

\*/

/\*

Load Sample Data into each of tables

Begin with the 'Type' and 'Value' tables created to manage data quality

\*/

----TYPE TABLES----

-- Insert Into Organization\_Type

INSERT INTO Organization\_Type([Org\_Type\_Value]) VALUES('Vendor');

INSERT INTO Organization\_Type([Org\_Type\_Value]) VALUES('Rescue');

INSERT INTO Organization\_Type([Org\_Type\_Value]) VALUES('Shelter');

INSERT INTO Organization\_Type([Org\_Type\_Value]) VALUES('Profit');

INSERT INTO Organization\_Type([Org\_Type\_Value]) VALUES('Partner');

-- Insert Into Event\_Type

INSERT INTO Event\_Type([Event\_Type\_Value]) VALUES('Adoption');

INSERT INTO Event\_Type([Event\_Type\_Value]) VALUES('Auction');

INSERT INTO Event\_Type([Event\_Type\_Value]) VALUES('Awareness');

INSERT INTO Event\_Type([Event\_Type\_Value]) VALUES('Donation Drive');

INSERT INTO Event\_Type([Event\_Type\_Value]) VALUES('Sales');

INSERT INTO Event\_Type([Event\_Type\_Value]) VALUES('Other');

-- Insert Into Countries

-- ISO Standard 2 ISO Country Code & Country Name imported from Excel

-- Insert Into Contact\_Sub\_Type\_Value

INSERT INTO Contact\_Sub\_Type\_Value([Contact\_Sub\_Type\_Value]) VALUES('Organization - Partner');

INSERT INTO Contact\_Sub\_Type\_Value([Contact\_Sub\_Type\_Value]) VALUES('Organization - Vendor');

INSERT INTO Contact\_Sub\_Type\_Value([Contact\_Sub\_Type\_Value]) VALUES('Personal - Donor');

INSERT INTO Contact\_Sub\_Type\_Value([Contact\_Sub\_Type\_Value]) VALUES('Personal - Employee');

INSERT INTO Contact\_Sub\_Type\_Value([Contact\_Sub\_Type\_Value]) VALUES('Personal - Volunteer');

INSERT INTO Contact\_Sub\_Type\_Value([Contact\_Sub\_Type\_Value]) VALUES('Personal - Foster');

INSERT INTO Contact\_Sub\_Type\_Value([Contact\_Sub\_Type\_Value]) VALUES('Personal - Adopter');

INSERT INTO Contact\_Sub\_Type\_Value([Contact\_Sub\_Type\_Value]) VALUES('Personal - Other');

-- Insert Into Contact\_Sub\_Type

-- Insert Into Contact\_Org\_Status

INSERT INTO Contact\_Org\_Status([Contact\_Org\_Status\_Value]) VALUES('Active');

INSERT INTO Contact\_Org\_Status([Contact\_Org\_Status\_Value]) VALUES('Inactive - Deceased');

INSERT INTO Contact\_Org\_Status([Contact\_Org\_Status\_Value]) VALUES('Inactive - Out of Business');

INSERT INTO Contact\_Org\_Status([Contact\_Org\_Status\_Value]) VALUES('Inactive - Other');

INSERT INTO Contact\_Org\_Status([Contact\_Org\_Status\_Value]) VALUES('Inactive - Bad Contact Data');

-- Insert Into Canine\_Location\_Type

INSERT INTO Canine\_Location\_Type([Canine\_Location\_Type\_Value]) VALUES('Shelter');

INSERT INTO Canine\_Location\_Type([Canine\_Location\_Type\_Value]) VALUES('Foster');

INSERT INTO Canine\_Location\_Type([Canine\_Location\_Type\_Value]) VALUES('Foster Hospice');

INSERT INTO Canine\_Location\_Type([Canine\_Location\_Type\_Value]) VALUES('Home');

INSERT INTO Canine\_Location\_Type([Canine\_Location\_Type\_Value]) VALUES('Recue');

-- Insert Into Canine\_Status

INSERT INTO Canine\_Status([Canine\_Status\_Value]) VALUES('Urgent Need');

INSERT INTO Canine\_Status([Canine\_Status\_Value]) VALUES('Need');

INSERT INTO Canine\_Status([Canine\_Status\_Value]) VALUES('Fostered');

INSERT INTO Canine\_Status([Canine\_Status\_Value]) VALUES('Sponsored');

INSERT INTO Canine\_Status([Canine\_Status\_Value]) VALUES('Adopted');

INSERT INTO Canine\_Status([Canine\_Status\_Value]) VALUES('Lost');

INSERT INTO Canine\_Status([Canine\_Status\_Value]) VALUES('Found');

INSERT INTO Canine\_Status([Canine\_Status\_Value]) VALUES('Reunited');

INSERT INTO Canine\_Status([Canine\_Status\_Value]) VALUES('Deceased');

-- Insert Into Cost\_Type

INSERT INTO Cost\_Type([Cost\_Type\_Value]) VALUES('Payroll');

INSERT INTO Cost\_Type([Cost\_Type\_Value]) VALUES('Supplies');

INSERT INTO Cost\_Type([Cost\_Type\_Value]) VALUES('Taxes');

INSERT INTO Cost\_Type([Cost\_Type\_Value]) VALUES('Veterinarian');

INSERT INTO Cost\_Type([Cost\_Type\_Value]) VALUES('Services');

INSERT INTO Cost\_Type([Cost\_Type\_Value]) VALUES('Other');

-- Insert Into Payment\_Type

INSERT INTO Payment\_Type([Payment\_Type\_Value]) VALUES('Adoption Fee');

INSERT INTO Payment\_Type([Payment\_Type\_Value]) VALUES('Sponsorship');

INSERT INTO Payment\_Type([Payment\_Type\_Value]) VALUES('Donation');

INSERT INTO Payment\_Type([Payment\_Type\_Value]) VALUES('Customer');

INSERT INTO Payment\_Type([Payment\_Type\_Value]) VALUES('Credit Adjustment');

INSERT INTO Payment\_Type([Payment\_Type\_Value]) VALUES('Other');

-- Insert Into Product\_Type

INSERT INTO Product\_Type([Product\_Type\_Value]) VALUES('Collar');

INSERT INTO Product\_Type([Product\_Type\_Value]) VALUES('Leash');

INSERT INTO Product\_Type([Product\_Type\_Value]) VALUES('Food Mat');

INSERT INTO Product\_Type([Product\_Type\_Value]) VALUES('Dog Bed Padding');

INSERT INTO Product\_Type([Product\_Type\_Value]) VALUES('Dog Bed Cover');

INSERT INTO Product\_Type([Product\_Type\_Value]) VALUES('Travel Water Bowl');

INSERT INTO Product\_Type([Product\_Type\_Value]) VALUES('Hiking Water Dispenser');

--Insert into Contact\_Organization

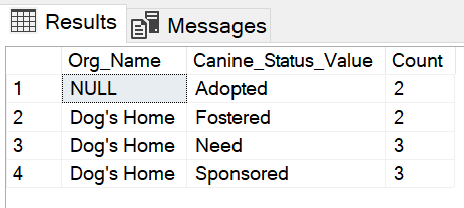
INSERT INTO Contact\_Organization([Contact\_Org\_Value]) VALUES('Contact');

INSERT INTO Contact\_Organization([Contact\_Org\_Value]) VALUES('Organization');

## Answers to Data Questions

The organization using the database in this example is Dog’s Home.

1. **What dogs are in the database, and what are their statuses?**



*Analysis – It looks like a Dog’s Home has two previous clients who have been adopted, and eight current clients that are fostered, are in need (not fostered or sponsored) or are sponsored.*

SELECT o.Org\_Name, cs.Canine\_Status\_Value, COUNT(c.Canine\_ID) as Count

FROM

Canines AS c

JOIN Canine\_Status AS cs

ON c.Canine\_Status\_ID=cs.Canine\_Status\_ID

LEFT JOIN Organizations AS o

ON c.Owner\_Org\_ID = o.Org\_ID

GROUP BY o.Org\_Name, cs.Canine\_Status\_Value;

1. **How much money was taken in a period?**



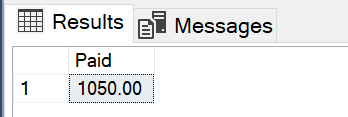
SELECT SUM(i.Amount\_Paid) as Payments

FROM Income AS i

WHERE Year(i.Paid\_Date)=2019

;

1. **How much money was spent in a period?**



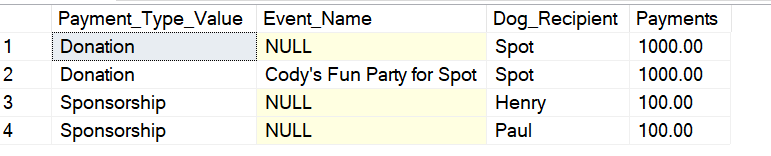
SELECT SUM(c.Amount\_Due) as Paid

FROM Costs AS c

WHERE Year(c.Paid\_Date)=2019

;

1. **What was the sources of the income?**



SELECT pt.Payment\_Type\_Value, fe.Event\_Name, ca.Current\_Name as Dog\_Recipient, SUM(i.Amount\_Paid) as Payments

FROM Income AS i

JOIN Payment\_Type AS pt

ON i.Payment\_Type\_ID=pt.Payment\_Type\_ID

LEFT JOIN Fun\_Events as fe

ON i.Event\_ID = fe.Event\_ID

LEFT JOIN Canines AS ca

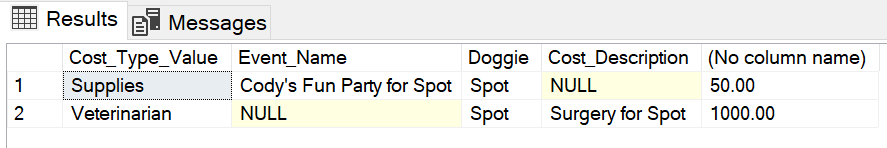
ON ca.Canine\_ID = i.Canine\_ID

WHERE Year(i.Paid\_Date)=2019

GROUP BY pt.Payment\_Type\_Value, fe.Event\_Name, ca.Current\_Name

;

1. **What were the details of the costs?**



SELECT ct.Cost\_Type\_Value, fe.Event\_Name, ca.Current\_Name as Doggie, co.Cost\_Description, SUM(co.Amount\_Due)

FROM Costs AS co

JOIN Cost\_Type AS ct

ON co.Cost\_Type\_ID=ct.Cost\_Type\_ID

LEFT JOIN Fun\_Events AS fe

ON co.Event\_ID = fe.Event\_ID

LEFT JOIN Canines AS ca

ON co.Canine\_ID = ca.Canine\_ID

WHERE Year(co.Paid\_Date)=2019

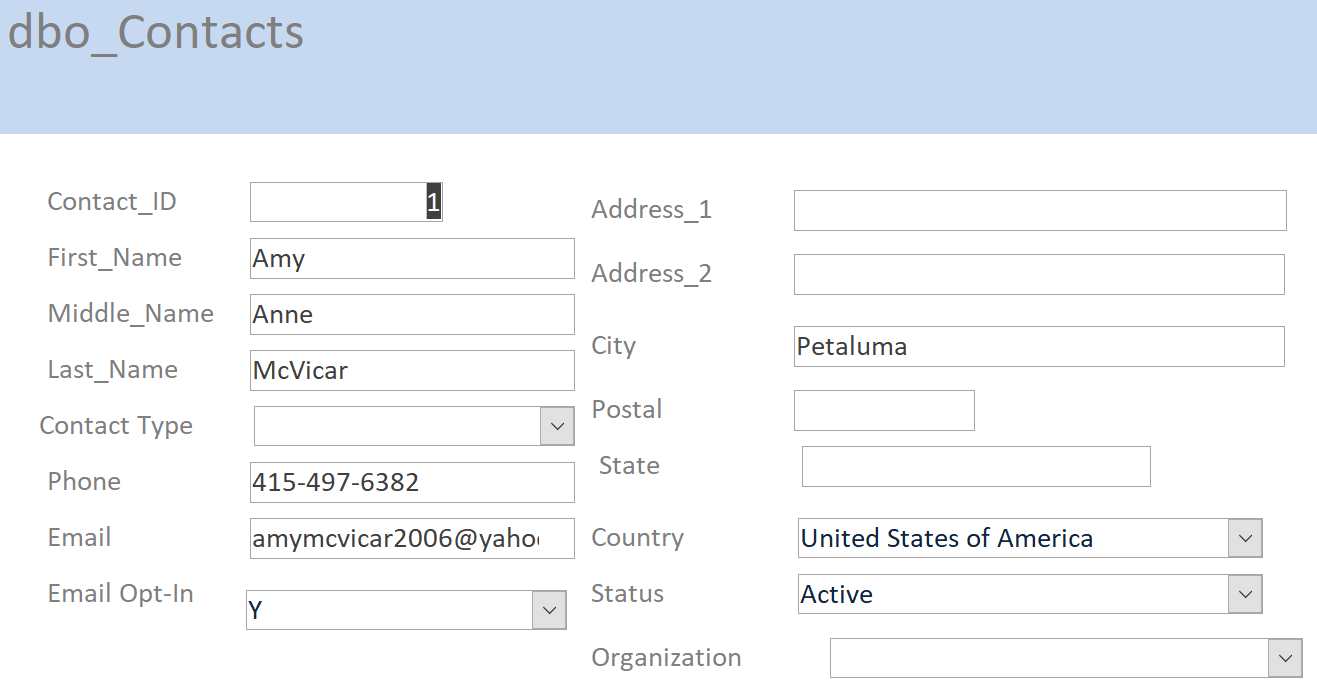
GROUP BY ct.Cost\_Type\_Value, fe.Event\_Name, ca.Current\_Name, co.Cost\_Description;

## GUI Prototype

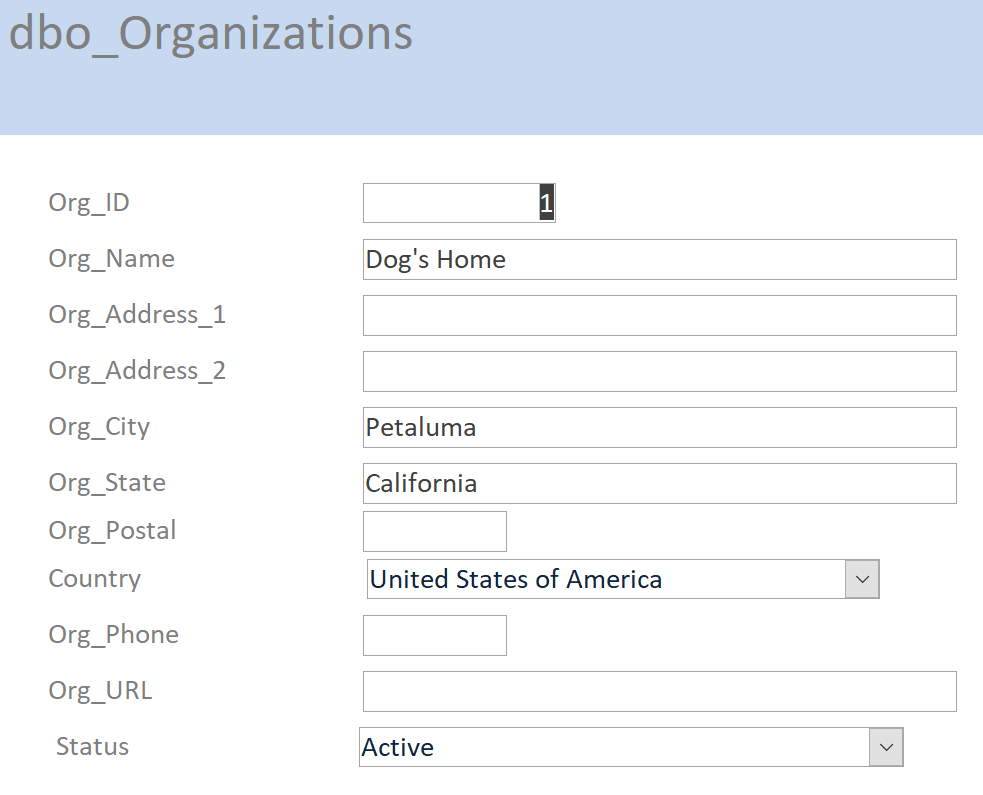
I created forms for entering data into the main objects:

* Contacts
* Organization
* Canines
* Events
* Costs
* Incomes

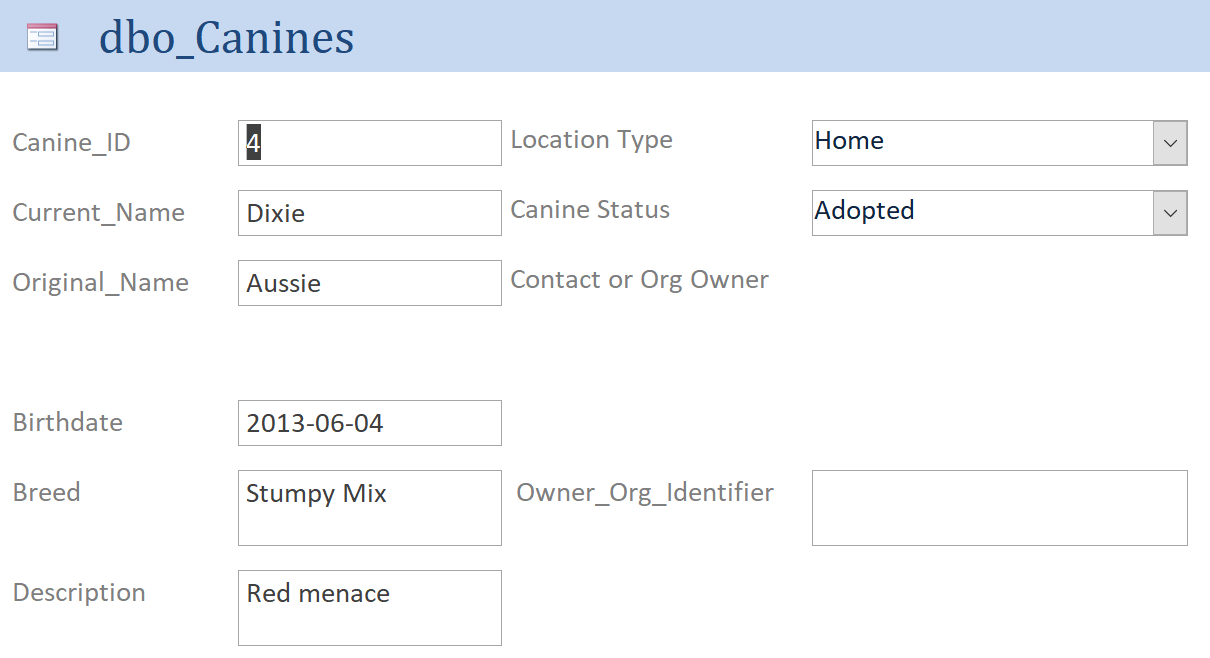
#### **Contacts**



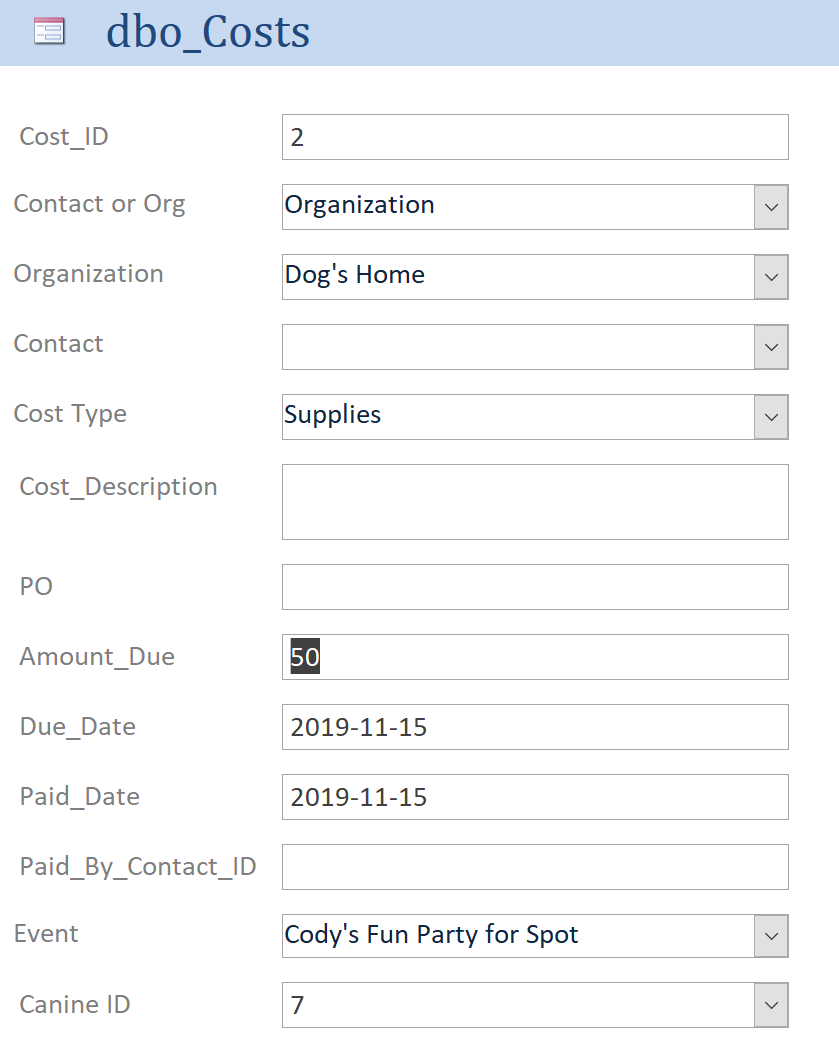
#### Organizations



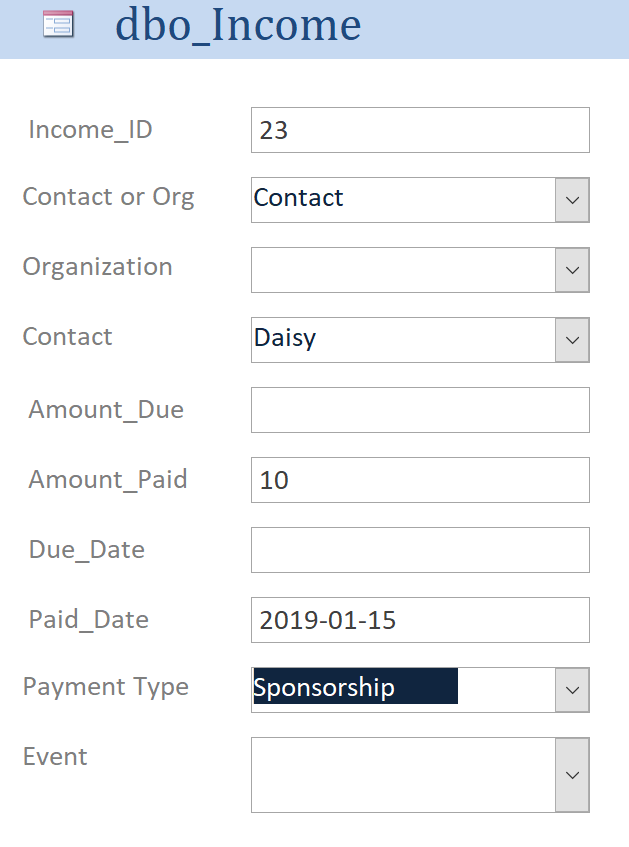
#### Canines



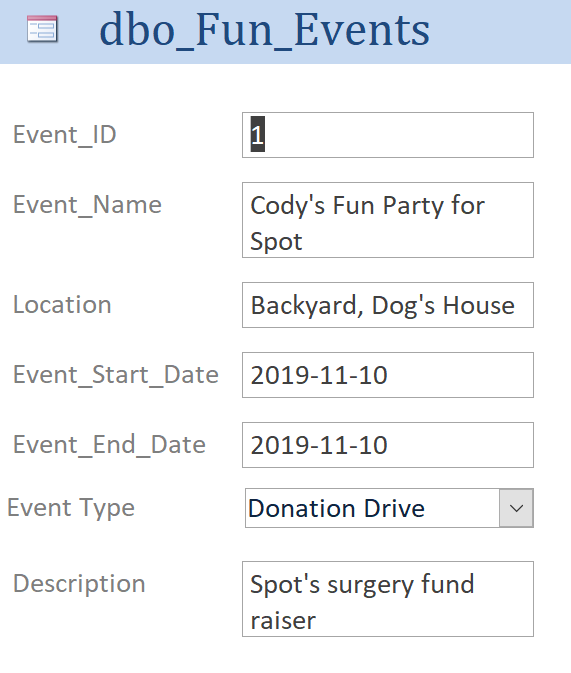
#### Costs



#### Income



#### Fun Events



## Reflection

I always knew that good design was the critical aspect of database design and usability, but I learned I underestimated the time needed. I learned that moving ahead before your design is ready costs more time and ultimately flaws in the final implementation. Another big take away for me from this class is the iterative and staged approach to design. In my false start, I made two mistakes: I included too much detail in the first stages and plans and jumped to the next steps prematurely. I didn’t understand starting with conceptual models that weren’t what you would know would be the logical and normalized versions, but now I see what a helpful and important step this is and why you focus on keeping it high level. My implementation is not perfect, and I see many improvements to make. I have two significant, extremely valuable outcomes from this class. One is a better idea on how to design this database in the future and the other is a new respect and understanding the design process which I will improve any of my efforts in time and end product.

