

Beta University Annual Fund

Tyler Paul, Evan Sturtevant, Ryan Hurlock

October 18, 2020

Table of Contents

Introduction.....	3
Home Page/Insert Form.....	4
Class Representative Report.....	5
Payments Due Report.....	6
Phonathon Volunteer Contact List.....	7
Annual Report.....	8
Final ER Diagram.....	11
DDL.....	12
Description Dumps.....	17
Triggers & Procedures	25
databaseForm.html	28
reports.php	34
donorUI.php.....	54
pledgeUI.php.....	58
donationUI.php	61
CSV Files.....	64
Archived Versions	65

Introduction

This database was designed to keep track of the pledges and reports for a University Annual Fund. It contains tables to keep track of donors who may donate or have donated in the past. They can keep track of those donor's employers who may match a donation provided by the donor. It especially allows for tracking the pledges and payments made to the university by said donors and organizations. The database automatically calculates and reports what donor circle the donor qualifies for based of the size of their donations.

Reporting features of the database include an annual report which provides the total amount gifted to the university for that particular year. The report will break that total down further based upon the category of donor, the graduation year of the donors, and what donor circle made the most money. Additionally, people who donated sufficiently to qualify for a donor circle will have their name printed under said circle. Other reports include the Payments due report which will list which donors pledged money to the university, but have yet to pay it; the Phonathon Volunteer Contact list, a list of potential donors with contact information and last years donation if any, so that a group of underpaid college students can harass them for money; and the Class Representative report which supplies relevant contact and donation information about each potential donor.

Access to the user interface is found through this link:

<http://jcsites.juniata.edu/students/sturtel19/databaseForm.html>

Home Page / Insert Form

Enter the following donor information (Where Applicable):

First Name*

Last Name*

Phone Number*

Email Address

Street Address*

Grad Year

Type of Donor

Matching Corporation

Zip Code*

Spouse Id

For Deletion: Donor Id

Enter the following pledge information (Where Applicable):

Pledge Date*

Anticipated Payment Date

Amount Pledged*

Donor ID*

Project Name

For Deletion: Pledge ID*

Enter the following donation information

Date Paid*

Amount Paid*

Payment Method*

Pledge ID*

For Deletion: Donation ID*

Class Representative Report:

REPORT:

Connected successfully

Name	Address	Phone	Last Year's Donation	This Year's Donation
Greg Harvey	521 Best Way Avenue, PA 16652	8145455667	6700.00	600.00
Steve France	548 Orange Road, PA 16652	1224556776		18000.00
Terry Owen	900 Clean House Avenue, PA 16652	6567879090	15000.00	13000.00
John Riley	435 Red Way St., PA 16652	4345450987		
Loren Rhodes	123 Street St, PA 16652	12345678901		100000.00
Evan Sturtevant	718 East Ave, PA 16652	12089202550	3300.00	100300.00
Josh Riley	222 Yellow Street, NY 12345	7678809922	1400.00	
George Green	323 Green House Road, NY 12345	8765432100	700.00	21000.00
Robert Geroge	234 StreetTown House, NY 12345	1235465684	5000.00	40000.00
Lance Armstrong	323 Deffrance Drive, NY 12345	6645232003		
Gerald Kruse	1700 Moore St, NY 12345	6666666666	200.00	500.00
Bill Thomas	1500 Sodor Isle, OH 44454	1111111111		
Tyler Paul	345 Street st, OH 44454	93875937594	50.00	575.00
John Eugene	435 Eugene Ave, MA 01864	3233233223		
Ryan Hurlock	68 Red Road, MA 01864	2222222222		

Payments Due Report:

REPORT:

Connected successfully

Name	Amount Due	Date Due	Address	Pledge	Amount Paid	Previous Payment Date
Evan Sturtevant	900000.00	2020-09-25	718 East Ave, PA 16652	1000000.00	100000.00	2020-09-23
Loren Rhodes	400000.00	2020-07-15	123 Street St, PA 16652	500000.00	100000.00	2020-06-30

Phonathon Volunteer Contact List

REPORT:

Connected successfully

Name	Address	Phone	Type of Donor	Grad Year	Last Year's Donation
Lance Armstrong	323 Defrance Drive, NY 12345	6645232003	OTHER	2001	
John Eugene	435 Eugene Ave, MA 01864	3233233223	OTHER		
Steve France	548 Orange Road, PA 16652	1224556776	OTHER	2001	
Robert Geroqe	234 StreetTown House, NY 12345	1235465684	OTHER	1998	5000.00
George Green	323 Green House Road, NY 12345	8765432100	ALUMNI	2001	700.00
Greg Harvey	521 Best Way Avenue, PA 16652	8145455667	PARENTS		6700.00
Ryan Hurlock	68 Red Road, MA 01864	2222222222	ALUMNI	1876	
Gerald Kruse	1700 Moore St, NY 12345	6666666666	PARENTS	1975	200.00
Terry Owen	900 Clean House Avenue, PA 16652	6567879090	ALUMNI	1998	15000.00
Tyler Paul	345 Street st, OH 44454	93875937594	PARENTS	1346	50.00
Loren Rhodes	123 Street St, PA 16652	12345678901	SENIOR	2001	
John Riley	435 Red Way St., PA 16652	4345450987	OTHER		
Josh Riley	222 Yellow Street, NY 12345	7678809922	ALUMNI	1998	1400.00
Evan Sturtevant	718 East Ave, PA 16652	12089202550	ALUMNI	1998	3300.00
Bill Thomas	1500 Sodor Isle, OH 44454	1111111111	OTHER	1442	

Annual Report

REPORT:

Connected successfully

Total Donated: 293400.00

By Category:

ALUMNI	134300.00
SENIOR	100000.00
OTHER	58000.00
PARENTS	1100.00

By Class:

Class of	Amount Donated	Percent Participated
1975	500.00	100.000
1998	133300.00	75.000
2001	139000.00	100.000

President's Circle: 200000.00

Class of	Amount Donated
1998	100000.00
2001	100000.00

Donors

Evan Sturtevant, Loren Rhodes,

Platinum Circle: 40000.00

Class of	Amount Donated
1998	40000.00

Donors

Robert Geroge,

Gold Circle: 21000.00

Gold I Circle: 21000.00

Class of	Amount Donated
2001	21000.00

Donors

George Green,

Gold II Circle: 18000.00

Class of	Amount Donated
2001	18000.00

Donors

Steve France,

Silver I Circle: 13000.00

Class of	Amount Donated
1998	13000.00

Donors

Terry Owen,

Silver II Circle:

Class of	Amount Donated
----------	----------------

Donors**Silver III Circle:**

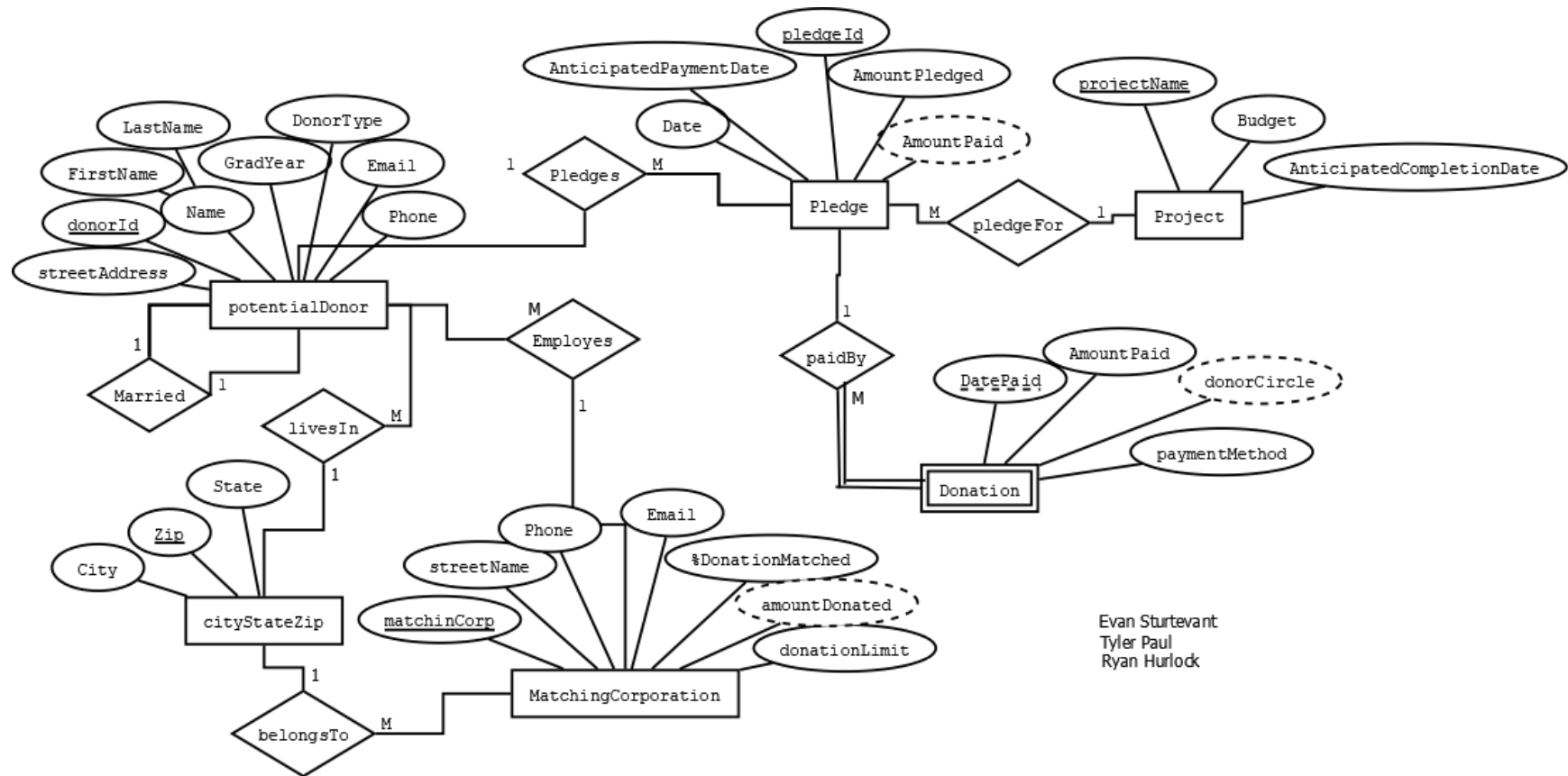
Class of	Amount Donated
----------	----------------

Donors**Bronze I Circle: 1100.00**

Class of	Amount Donated
1975	500.00

Bronze I Circle: 1100.00	Class of		Amount Donated
	1975	500.00	
		600.00	
	Donors		
	Gerald Kruse, Greg Harvey,		
Bronze II Circle: 300.00	Class of		Amount Donated
	1998	300.00	
	Donors		
	Evan Sturtevant,		
Bronze III Circle:	Class of		Amount Donated
	Donors		

Final ER Diagram:



DDL

```
--Team: Evan Sturtevant, Tyler Paul, Ryan Hurlock
```

```
--Date 11/20/2020
```

```
DROP TABLE donation;
```

```
DROP TABLE pledge;
```

```
DROP TABLE potentialDonor;
```

```
DROP TABLE matchingCorporation;
```

```
DROP TABLE cityStateZip;
```

```
DROP TABLE project;
```

```
CREATE TABLE project
```

```
(projectName      CHAR      (20)      NOT NULL UNIQUE,
```

```
budget            NUMERIC (11,2),
```

```
compDate          DATE,
```

```
PRIMARY KEY (projectName)
```

```
CHECK (compDate > current_date);
```

```
);
```

```
CREATE TABLE cityStateZip
```

```
(zip      CHAR      (5)      NOT NULL UNIQUE,
```

```
city      VARCHAR (20)      NOT NULL,
```

```
state     char      (2)      NOT NULL,
```

```
PRIMARY KEY (zip)
```

```
);
```

```

CREATE TABLE matchingCorporation
(matchingCorp      CHAR      (20),
streetAddress      VARCHAR   (50),
phone              CHAR      (11),
email              VARCHAR   (50),
percentMatched     NUMERIC   (5,2),
donationLimit      NUMERIC   (11,2),
zip CHAR (5),

```

```

PRIMARY KEY (matchingCorp),
FOREIGN KEY (zip)
    REFERENCES cityStateZip (zip),
CHECK (email LIKE '%_@_%._%')
);

```

```

CREATE TABLE potentialDonor
(donorId           SERIAL,
firstName          VARCHAR   (20)      NOT NULL,
lastName           VARCHAR   (20)      NOT NULL,
phone              CHAR      (11)      NOT NULL,
email              VARCHAR   (50),
streetAddress      VARCHAR   (50)      NOT NULL,
gradYear           CHAR      (4),
donorType           CHAR      (10),
matchingCorp       CHAR      (20),
zip                CHAR      (5)      NOT NULL,
spouseId           INTEGER,

```

```

PRIMARY KEY (donorId),
FOREIGN KEY (matchingCorp)
    REFERENCES matchingCorporation (matchingCorp),
FOREIGN KEY (zip)
    REFERENCES cityStateZip (zip),
FOREIGN KEY (spouseId)
    REFERENCES potentialDonor (donorId),
CHECK (donorType IN ('SENIOR', 'ALUMNI', 'PARENTS', 'FACULTY', 'ADMINISTRATOR', 'STAFF',
    'CORPORATIONS', 'OTHER')),
CHECK (email LIKE '%_@_%._%')
);

```

```

CREATE TABLE pledge
(pledgeId          SERIAL,
date              DATE          NOT NULL,
anticipatedPaymentDate DATE,
amountPledged     NUMERIC (11,2) NOT NULL,
donorId           INTEGER       NOT NULL,
projectName       char          (20),
amountPaid        NUMERIC (11,2),

```

```

PRIMARY KEY (pledgeId),
FOREIGN KEY (donorId)
    REFERENCES potentialDonor (donorId),
FOREIGN KEY (projectName)
    REFERENCES project (projectName)
CHECK (date < anticipatedPaymentDate);
);

```

```

CREATE TABLE donation

```

```

(datePaid      DATE          NOT NULL,
amountPaid     NUMERIC (11,2) NOT NULL,
paymentMethod  CHAR         (6)      NOT NULL,
pledgeId       INTEGER       NOT NULL,
donorCircle    CHAR         (15),

PRIMARY KEY (datePaid, pledgeId),
FOREIGN KEY (pledgeId)
    REFERENCES pledge (pledgeId),
CHECK (paymentMethod in('CREDIT', 'DEBIT', 'CASH', 'CHECK'))
);

--Views

CREATE VIEW lastYearPaid (donorId, totalPaid)
AS SELECT p.donorID, sum(d.amountPaid)
FROM potentialDonor p, pledge i, donation d
WHERE p.donorId = i.donorId AND i.pledgeId = d.pledgeId
AND date_part('year', d.datePaid) = date_part('year', CURRENT_DATE) -1
GROUP BY p.donorId;

CREATE VIEW thisYearPaid (donorId, totalPaid)
AS SELECT p.donorID, sum(d.amountPaid)
FROM potentialDonor p, pledge i, donation d
WHERE p.donorId = i.donorId AND i.pledgeId = d.pledgeId
AND date_part('year', d.datePaid) = date_part('year', CURRENT_DATE)
GROUP BY p.donorId;

```

```

CREATE VIEW phonathonVolunteer (Name, Address, Phone, donorType, gradYear, lastYearPayment)
AS SELECT concat(p.firstName, ' ', p.lastName), concat(p.streetAddress, ', ', z.state, ' ',
p.zip),
p.phone, p.donorType, p.gradYear, l.totalPaid
FROM potentialDonor p LEFT JOIN lastYearPaid l on p.donorId = l.donorId,
cityStateZip z
WHERE z.zip = p.zip
ORDER BY p.lastName;

```

```

CREATE VIEW classRepresentative (Name, Address, Phone, lastYearPayment, thisYearPayment)
AS SELECT concat(p.firstName, ' ', p.lastName), concat(p.streetAddress, ', ', z.state, ' ',
p.zip),
p.phone, l.totalPaid, t.totalPaid
FROM potentialDonor p LEFT JOIN lastYearPaid l on p.donorId = l.donorId
LEFT JOIN thisYearPaid t on p.donorId = t.donorId , cityStateZip z
WHERE z.zip = p.zip;

```

```

CREATE VIEW paymentsDue (Name, AmountDue, DueDate, Address, Pledge, AmountPaid,
PreviousPaymentDate)
AS SELECT concat(d.firstName, ' ', d.lastName), (p.amountPledged - p.amountPaid),
p.anticipatedPaymentDate,
concat(d.streetAddress, ', ', z.state, ' ', d.zip), p.amountPledged, p.amountPaid, (SELECT
max(donation.datepaid) from donation where p.pledgeId = donation.pledgeId)
FROM potentialDonor d NATURAL JOIN pledge p NATURAL JOIN cityStateZip z
WHERE p.amountPaid < p.amountPledged
order by p.anticipatedPaymentDate DESC;

```

```

CREATE VIEW circleNames (Circle, Name, Class, Amount, Date)
AS SELECT d.donorCircle, concat(p.firstName, ' ', p.lastName), p.gradYear, d.amountPaid,
d.datePaid

```



```
FROM potentialDonor p NATURAL JOIN pledge NATURAL JOIN donation d;
```

```
CREATE VIEW totalSummary(amount, class, category, date, donorId)
AS SELECT d.amountPaid, p.gradYear, p.donorType, d.datepaid, p.donorId
FROM potentialDonor p natural join pledge natural join donation d;
```

Description Dumps

List of relations			
Schema	Name	Type	Owner
public	circlenames	view	sturtell19
public	citystatezip	table	sturtell19
public	classrepresentative	view	sturtell19
public	donation	table	sturtell19
public	lastyearpaid	view	sturtell19
public	matchingcorporation	table	sturtell19
public	paymentsdue	view	sturtell19
public	phonathonvolunteer	view	sturtell19
public	pledge	table	sturtell19
public	pledge_pledgeid_seq	sequence	sturtell19
public	potentialdonor	table	sturtell19
public	potentialdonor_donorid_seq	sequence	sturtell19
public	project	table	sturtell19
public	thisyearpaid	view	sturtell19
public	totalsummary	view	sturtell19

Table "public.potentialdonor"

Column	Type	Modifiers
donorid	integer	not null default
nextval('potentialdonor_donorid_seq'::regclass)		
firstname	character varying(20)	not null
lastname	character varying(20)	not null
phone	character(11)	not null
email	character varying(50)	
streetaddress	character varying(50)	not null
gradyear	character(4)	
donortype	character(10)	
matchingcorp	character(20)	
zip	character(5)	not null
spouseid	integer	

Indexes:

"potentialdonor_pkey" PRIMARY KEY, btree (donorid)

Check constraints:

"potentialdonor_donortype_check" CHECK (donortype = ANY (ARRAY['SENIOR'::bpchar, 'ALUMNI'::bpchar, 'PARENTS'::bpchar, 'FACULTY'::bpchar, 'ADMINISTRATOR'::bpchar, 'STAFF'::bpchar, 'CORPORATIONS'::bpchar, 'OTHER'::bpchar]))

"potentialdonor_email_check" CHECK (email::text ~~ '%_@_%._%'::text)

Foreign-key constraints:

"potentialdonor_matchingcorp_fkey" FOREIGN KEY (matchingcorp) REFERENCES matchingcorporation(matchingcorp)

"potentialdonor_spouseid_fkey" FOREIGN KEY (spouseid) REFERENCES potentialdonor(donorid)

"potentialdonor_zip_fkey" FOREIGN KEY (zip) REFERENCES citystatezip(zip)

Referenced by:

TABLE "pledge" CONSTRAINT "pledge_donorid_fkey" FOREIGN KEY (donorid) REFERENCES potentialdonor(donorid)

TABLE "potentialdonor" CONSTRAINT "potentialdonor_spouseid_fkey" FOREIGN KEY (spouseid) REFERENCES potentialdonor(donorid)

Triggers:

updatetype BEFORE INSERT OR UPDATE ON potentialdonor FOR EACH ROW EXECUTE PROCEDURE updatetype()

Table "public.pledge"

Column	Type	Modifiers
pledgeid	integer	not null default
nextval('pledge_pledgeid_seq'::regclass)		
date	date	not null
anticipatedpaymentdate	date	
amountpledged	numeric(11,2)	not null
donorid	integer	not null
projectname	character(20)	
amountpaid	numeric(11,2)	

Indexes:

"pledge_pkey" PRIMARY KEY, btree (pledgeid)

Check constraints:

"pledge_check" CHECK (date < anticipatedpaymentdate)

Foreign-key constraints:

"pledge_donorid_fkey" FOREIGN KEY (donorid) REFERENCES potentialdonor(donorid)

"pledge_projectname_fkey" FOREIGN KEY (projectname) REFERENCES project(projectname)

Referenced by:

```
TABLE "donation" CONSTRAINT "donation_pledgeid_fkey" FOREIGN KEY (pledgeid) REFERENCES
pledge(pledgeid)
```

Table "public.donation"

Column	Type	Modifiers
-----+-----+-----		
datepaid	date	not null
amountpaid	numeric(8,2)	not null
paymentmethod	character(6)	not null
pledgeid	integer	not null
donorcircle	character(15)	

Indexes:

```
"donation_pkey" PRIMARY KEY, btree (datepaid, pledgeid)
```

Check constraints:

```
"donation_paymentmethod_check" CHECK (paymentmethod = ANY (ARRAY['Credit'::bpchar,
'Debit'::bpchar, 'Cash'::bpchar, 'Check'::bpchar]))
```

Foreign-key constraints:

```
"donation_pledgeid_fkey" FOREIGN KEY (pledgeid) REFERENCES pledge(pledgeid)
```

Triggers:

```
updatecircle BEFORE INSERT OR UPDATE ON donation FOR EACH ROW EXECUTE PROCEDURE
```

```
updatecircle()
```

```
updatepledge AFTER INSERT OR DELETE OR UPDATE ON donation FOR EACH STATEMENT EXECUTE
PROCEDURE updatepledge()
```

Table "public.citystatezip"

Column	Type	Modifiers
zip	character(5)	not null
city	character varying(20)	not null
state	character(2)	not null

Indexes:

"citystatezip_pkey" PRIMARY KEY, btree (zip)

Referenced by:

TABLE "matchingcorporation" CONSTRAINT "matchingcorporation_zip_fkey" FOREIGN KEY (zip) REFERENCES citystatezip(zip)

TABLE "potentialdonor" CONSTRAINT "potentialdonor_zip_fkey" FOREIGN KEY (zip) REFERENCES citystatezip(zip)

Table "public.project"

Column	Type	Modifiers
projectname	character(20)	not null
budget	numeric(11,2)	
comdate	date	

Indexes:

"project_pkey" PRIMARY KEY, btree (projectname)

Check constraints:

"project_comdate_check" CHECK (comdate > 'now'::text::date)

Referenced by:

```
TABLE "pledge" CONSTRAINT "pledge_projectname_fkey" FOREIGN KEY (projectname) REFERENCES
project(projectname)
```

Table "public.matchingcorporation"

Column	Type	Modifiers
matchingcorp	character(20)	not null
streetaddress	character varying(50)	
phone	character(11)	
email	character varying(50)	
percentmatched	numeric(5,2)	
donationlimit	numeric(11,2)	
zip	character(5)	

Indexes:

```
"matchingcorporation_pkey" PRIMARY KEY, btree (matchingcorp)
```

Check constraints:

```
"matchingcorporation_email_check" CHECK (email::text ~~ '%_@%._%':text)
```

Foreign-key constraints:

```
"matchingcorporation_zip_fkey" FOREIGN KEY (zip) REFERENCES citystatezip(zip)
```

Referenced by:

```
TABLE "potentialdonor" CONSTRAINT "potentialdonor_matchingcorp_fkey" FOREIGN KEY
(matchingcorp) REFERENCES matchingcorporation(matchingcorp)
```

View "public.totalsummary"

Column	Type	Modifiers
amount	numeric(8,2)	
class	character(4)	
category	character(10)	

date	date	
donorid	integer	

View "public.circlenames"

Column	Type	Modifiers
-----+-----+-----		
circle	character(15)	
name	text	
class	character(4)	
amount	numeric(8,2)	
date	date	

View "public.phonathonvolunteer"

Column	Type	Modifiers
-----+-----+-----		
name	text	
address	text	
phone	character(11)	
donortype	character(10)	
gradyear	character(4)	
lastyearpayment	numeric	

View "public.classrepresentative"

Column	Type	Modifiers
-----+-----+-----		
name	text	
address	text	
phone	character(11)	
lastyearpayment	numeric	

```
thisyearpayment | numeric      |
```

View "public.paymentsdue"

Column	Type	Modifiers
name	text	
amountdue	numeric	
duedate	date	
address	text	
pledge	numeric(11,2)	
amountpaid	numeric(11,2)	
previouspaymentdate	date	

View "public.thisyearpaid"

Column	Type	Modifiers
donorid	integer	
totalpaid	numeric	

View "public.lastyearpaid"

Column	Type	Modifiers
donorid	integer	
totalpaid	numeric	

Triggers & Procedures

--This trigger updates the amountPaid for a pledge whenever a donation is made towards it

```
CREATE FUNCTION OR REPLACE updatePledge()
```

```
RETURNS trigger as $$
```

```
BEGIN
```

```
    UPDATE pledge
```

```
        SET amountPaid = (SELECT Sum(amountPaid) FROM donation d WHERE d.pledgeId =
pledge.pledgeId);
```

```
    RETURN NULL;
```

```
END;
```

```
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER updatePledge
```

```
AFTER UPDATE OR DELETE OR INSERT
```

```
ON donation FOR STATEMENT
```

```
EXECUTE PROCEDURE updatePledge();
```

--Updates the circle based on the size of the donation

```
CREATE OR REPLACE FUNCTION updateCircle()
```

```
RETURNS trigger as $$
```

```
BEGIN
```

```
    IF new.amountPaid > 49999.99 THEN
```

```
        new.donorCircle := 'Presidents';
```

```
    elsif new.amountPaid > 24999.99 THEN
```

```
        new.donorCircle := 'Platinum';
```

```

elsif new.amountPaid > 19999.99 THEN
    new.donorCircle := 'Gold I';
elsif new.amountPaid > 14999.99 THEN
    new.donorCircle := 'Gold II';
elsif new.amountPaid > 9999.99 THEN
    new.donorCircle := 'Silver I';
elsif new.amountPaid > 4999.99 THEN
    new.donorCircle := 'Silver II';
elsif new.amountPaid > 999.99 THEN
    new.donorCircle := 'Silver III';
elsif new.amountPaid > 499.99 THEN
    new.donorCircle := 'Bronze I';
elsif new.amountPaid > 249.99 THEN
    new.donorCircle := 'Bronze II';
elsif new.amountPaid > 99.99 THEN
    new.donorCircle := 'Bronze III';
END IF;

```

```

RETURN NEW;

```

```

END;

```

```

$$ LANGUAGE plpgsql;

```

```

--

```

```

CREATE TRIGGER updateCircle
BEFORE UPDATE OR INSERT
ON donation FOR EACH ROW
EXECUTE PROCEDURE updateCircle();

```

```
--Update what type of donor in the case of empty entry
CREATE OR REPLACE FUNCTION updateType()
RETURNS TRIGGER AS $$
BEGIN

    IF new.donorType IS NULL THEN
        new.donorType := 'OTHER';
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER updateType
BEFORE UPDATE OR INSERT
ON potentialDonor FOR EACH ROW
EXECUTE PROCEDURE updateType();
```

databaseForm.HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>Uni Database Form</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<style>
body{
    background-color: #1d2d50;
    color: white;
    text-align: center;
}

</style>

<body>
<div id="formContainer">
    <form name="donorForm" method="GET"
action="http://jcsites.juniata.edu/students/sturtell19/donorUI.php">
    Enter the following donor information (Where Applicable):
    <p>
    <label>First Name*
    <input type="text" name="firstname">
```

```
</label>
<br>
<label>Last Name*
<input type="text" name="lastname">
</label>
<br>
<label>Phone Number*
<input type="text" name="phone">
</label>
<br>
<label>Email Address
<input type="text" name="email">
</label>
<br>
<label>Street Address*
<input type="text" name="street">
</label>
<br>
<label>Grad Year
<input type="text" name="grad">
</label>
<br>
<label>Type of Donor
<input type="text" name="donor">
</label>
<br>
<label>Matching Corporation
<input type="text" name="corp">
</label>
<br>
```

```

<label>Zip Code*
<input type="text" name="zip">
</label>
<br>
<label>Spouse Id
<input type="text" name="spouse">
</label>

<br><br>
For Deletion:
<label>Donor Id
<input type="text" name="donorId">
</label>
</p>
<p>
<input type="submit" name="Submit" value="Submit">
<input type="submit" name="Delete" value="Delete">
<label> </label>
</p>
</form>

```

```

<form name="pledgeForm" method="GET"
action="http://jcsites.juniata.edu/students/sturtell9/pledgeUI.php">
  Enter the following pledge information (Where Applicable):
  <p>
    <label>Pledge Date*
    <input type="text" name="pledgeDate">
    </label>
    <br>
    <label>Anticipated Payment Date

```

```

        <input type="text" name="antPayDate">
    </label>
    <br>
    <label>Amount Pledged*
    <input type="text" name="amountPledged">
    </label>
    <br>
    <label>Donor ID*
    <input type="text" name="donorId">
    </label>
    <br>
    <label>Project Name
    <input type="text" name="projName">
    </label>

    <br><br>
    For Deletion:
    <label>Pledge ID*
    <input type="text" name="pledgeId">
    </label>
</p>
<p>
        <input type="submit" name="Submit" value="Submit">
        <input type="submit" name="Delete" value="Delete">
    </p>
</form>

<form name="donationForm" method="GET"
action="http://jcsites.juniata.edu/students/sturtell19/donationUI.php">
    Enter the following donation information

```

```

<p>
  <label> Date Paid*
  <input type="text" name="datePaid">
</label>
  <br>
  <label> Amount Paid*
  <input type="text" name="amountPaid">
</label>
  <br>
  <label> Payment Method*
  <input type="text" name="method">
</label>
  <br>
  <label> Pledge ID*
  <input type="text" name="pledgeId">
</label>

  <br><br>
  For Deletion:
  <label>Donation ID*
  <input type="text" name="donationId">
</label>
</p>
<p>
  <input type="submit" name="Submit" value="Submit">
  <input type="submit" name="Delete" value="Delete">
</p>
</form>
</div>
<div id="reports">

```



```
<form name="phonathon" method="GET"
action="http://jcsites.juniata.edu/students/sturtell9/reports.php">
  <input type="submit" name="PhonathonReport" value="PhonathonReport">
  <input type="submit" name="ClassRepresentitive" value="ClassRepresentitive">
  <input type="submit" name="PaymentsDue" value="PaymentsDue">
  <input type="submit" name="AnnualReport" value="AnnualReport">
</form>

</div>
<p>&nbsp; </p>
</body>
</html>
```

Report.php

```
<html>

<head>
<meta name="GENERATOR" content="Microsoft FrontPage 5.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<title>Report</title>
</head>

<style>
body{
    background-color: #1d2d50;
    color: white;
    text-align:center;
}

table{
    margin-top: 7px;
    margin-left: auto;
    margin-right: auto;
}

hr{
    color: white;
    background-color: white;
```

```

}

td{
    padding: 5px;
}

</style>

<body>
<h1><center> REPORT: </center></h1>
<?php
    error_reporting(0);
    $link = pg_connect("host=itcsdbms user=sturtell19 password=June2001231 dbname=sampleels")
        or die ("Could not connect to database ");
    print ("Connected successfully\n");

    if ($_GET['PhonathonReport']) {
        $query = "SELECT * FROM phonathonvolunteer; ";
        $result = pg_query ($query)
            or die ("Query failed");

        print "<hr><table>\n";
        print "\t<tr>\n";
        print "\t\t<td>Name</td>\n";
        print "\t\t<td>Address</td>\n";
        print "\t\t<td>Phone</td>\n";
        print "\t\t<td>Type of Donor</td>\n";
        print "\t\t<td>Grad Year</td>\n";
        print "\t\t<td>Last Year's Donation</td>\n";
        print "\t</tr>\n";
    }
}

```

```

while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    print "\t<tr>\n";
    while(list($col_name, $col_value) = each($line)){
        print "\t\t<td>$col_value</td>\n";
    }
    print "\t</tr>\n";
}
print "</table>\n";
}
else if ($_GET['ClassRepresentative']) {
    $query = "SELECT * FROM classrepresentative; ";
    $result = pg_query ($query)
        or die ("Query failed");

    print "<hr><table>\n";
    print "\t<tr>\n";
    print "\t\t<td>Name</td>\n";
    print "\t\t<td>Address</td>\n";
    print "\t\t<td>Phone</td>\n";
    print "\t\t<td>Last Year's Donation</td>\n";
    print "\t\t<td>This Year's Donation</td>\n";
    print "\t</tr>\n";

    while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
        print "\t<tr>\n";
        while(list($col_name, $col_value) = each($line)){
            print "\t\t<td>$col_value</td>\n";
        }
        print "\t</tr>\n";
    }
}

```

```

    }
    print "</table>\n";

}
else if ($_GET['PaymentsDue']) {
    $query = "SELECT * FROM paymentsdue; ";
    $result = pg_query ($query)
        or die ("Query failed");

    print "<hr><table>\n";
    print "\t<tr>\n";
    print "\t\t<td>Name</td>\n";
    print "\t\t<td>Amount Due</td>\n";
    print "\t\t<td>Date Due</td>\n";
    print "\t\t<td>Address</td>\n";
    print "\t\t<td>Pledge</td>\n";
    print "\t\t<td>Amount Paid</td>\n";
    print "\t\t<td>Previous Payment Date</td>\n";
    print "\t</tr>\n";

    while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
        print "\t<tr>\n";
        while(list($col_name, $col_value) = each($line)){
            print "\t\t<td>$col_value</td>\n";
        }
        print "\t</tr>\n";
    }
    print "</table>\n";
}

```

```

else if ($_GET['AnnualReport']) {

    #Total Donated
    $query = " SELECT SUM(amount) FROM totalsummary WHERE date_part('year', date) =
date_part('year', CURRENT_DATE);";
    $result = pg_query ($query)
        or die ("Query failed");
    $totalArray = pg_fetch_array($result, NULL, PGSQL_ASSOC);
    list($header, $total) = each($totalArray);
    print "<h1 style=\"text-align:left\"> Total Donated: $total </h1>";

    #By Category
    $query = "SELECT category, sum(amount) as total FROM totalsummary WHERE date_part('year',
date) = date_part('year', CURRENT_DATE) GROUP BY category ORDER BY total DESC;";
    $result = pg_query ($query)
        or die ("Query failed");
    print "<hr><h2 style=\"text-align:left\">By Category: </h2>";
    print "<table>\n";
    while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
        print "\t<tr>\n";
        while(list($col_name, $col_value) = each($line)){
            print "\t\t<td>$col_value</td>\n";
        }
        print "\t</tr>\n";
    }
    print "</table>\n";

    #By Class
    $query = "SELECT s.class, sum(s.amount) as TOTAL,

```

```

        CAST( (100 * ((count(DISTINCT s.donorId) * 1.0) / count(p.donorId))) as DECIMAL (6,3)) as
PROPORTION
FROM potentialDonor p left join totalsummary s on s.donorid = p.donorid
WHERE (date_part('year', s.date) = date_part('year', CURRENT_DATE) or date is null)
AND p.gradYear IS NOT NULL AND s.class IS NOT NULL group by s.class;";
$result = pg_query ($query)
        or die ("Query failed");
print "<hr><h2 style=\"text-align:left\">By Class: </h2>";
print "<table>\n";
print "\t<tr>\n";
print "\t\t<td>Class of</td>\n";
print "\t\t<td>Amount Donated</td>\n";
print "\t\t<td>Percent Participated</td>\n";
print "\t</tr>\n";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    print "\t<tr>\n";
    while(list($col_name, $col_value) = each($line)){
        print "\t\t<td>$col_value</td>\n";
    }
    print "\t</tr>\n";
}
print "</table>\n";

#Pres
$query = " SELECT SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Presidents';";
$result = pg_query ($query)
        or die ("Query failed");
$totalArray = pg_fetch_array($result, NULL, PGSQL_ASSOC);
list($header, $total) = each($totalArray);

```

```

print "<hr><h3 style=\"text-align:left\"> President's Circle: $total </h3>";

$query = " SELECT DISTINCT class, SUM(amount) FROM circlenames WHERE date_part('year',
date) = date_part('year', CURRENT_DATE) AND circle = 'Presidents' group by class;";
$result = pg_query ($query)
    or die ("Query failed");
print "<table>\n";
print "\t<tr>\n";
print "\t\t<td>Class of</td>\n";
print "\t\t<td>Amount Donated</td>\n";
print "\t</tr>\n";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    print "\t<tr>\n";
    while(list($col_name, $col_value) = each($line)){
        print "\t\t<td>$col_value</td>\n";
    }
    print "\t</tr>\n";
}
print "</table>\n";

$query = " SELECT DISTINCT name from circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Presidents';";
$result = pg_query ($query)
    or die ("Query failed");
print "<h3> Donors </h3>";
print "<p>";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    while(list($col_name, $col_value) = each($line)){
        print "$col_value, ";
    }
}

```



```

}
print "</p>";

#Platinum
$query = " SELECT SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Platinum';";
$result = pg_query ($query)
    or die ("Query failed");
$totalArray = pg_fetch_array($result, NULL, PGSQL_ASSOC);
list($header, $total) = each($totalArray);
print "<hr><h3 style=\"text-align:left\"> Platinum Circle: $total </h3>";

$query = " SELECT class, SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Platinum' group by class;";
$result = pg_query ($query)
    or die ("Query failed");
print "<table>\n";
print "\t<tr>\n";
print "\t\t<td>Class of</td>\n";
print "\t\t<td>Amount Donated</td>\n";
print "\t</tr>\n";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    print "\t<tr>\n";
    while(list($col_name, $col_value) = each($line)){
        print "\t\t<td>$col_value</td>\n";
    }
    print "\t</tr>\n";
}
print "</table>\n";

```

```

$query = " SELECT DISTINCT name from circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Platinum';";
$result = pg_query ($query)
    or die ("Query failed");
print "<h3> Donors </h3>";
print "<p>";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    while(list($col_name, $col_value) = each($line)){
        print "$col_value, ";
    }
}
print "</p>";

#Gold I
$query = " SELECT SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Gold I';";
$result = pg_query ($query)
    or die ("Query failed");
$totalArray = pg_fetch_array($result, NULL, PGSQL_ASSOC);
list($header, $total) = each($totalArray);
print "<hr><h3 style=\"text-align:left\"> Gold I Circle: $total </h3>";

$query = " SELECT class, SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Gold I' group by class;";
$result = pg_query ($query)
    or die ("Query failed");
print "<table>\n";
print "\t<tr>\n";
print "\t\t<td>Class of</td>\n";
print "\t\t<td>Amount Donated</td>\n";

```

```

print "\t</tr>\n";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    print "\t<tr>\n";
    while(list($col_name, $col_value) = each($line)){
        print "\t\t<td>$col_value</td>\n";
    }
    print "\t</tr>\n";
}
print "</table>\n";

```

```

$query = " SELECT DISTINCT name from circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Gold I';";
$result = pg_query ($query)
    or die ("Query failed");
print "<h3> Donors </h3>";
print "<p>";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    while(list($col_name, $col_value) = each($line)){
        print "$col_value, ";
    }
}
print "</p>";

```

```

        #Gold II
$query = " SELECT SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Gold II';";
$result = pg_query ($query)
    or die ("Query failed");
$totalArray = pg_fetch_array($result, NULL, PGSQL_ASSOC);
list($header, $total) = each($totalArray);

```

```

print "<hr><h3 style=\"text-align:left\"> Gold II Circle: $total </h3>";

$query = " SELECT class, SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Gold II' group by class;";
$result = pg_query ($query)
    or die ("Query failed");
print "<table>\n";
print "\t<tr>\n";
print "\t\t<td>Class of</td>\n";
print "\t\t<td>Amount Donated</td>\n";
print "\t</tr>\n";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    print "\t<tr>\n";
    while(list($col_name, $col_value) = each($line)){
        print "\t\t<td>$col_value</td>\n";
    }
    print "\t</tr>\n";
}
print "</table>\n";

$query = " SELECT DISTINCT name from circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Gold II';";
$result = pg_query ($query)
    or die ("Query failed");
print "<h3> Donors </h3>";
print "<p>";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    while(list($col_name, $col_value) = each($line)){
        print "$col_value, ";
    }
}

```

```

}
print "</p>";

        #Silver I
        $query = " SELECT SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Silver I';";
        $result = pg_query ($query)
            or die ("Query failed");
        $totalArray = pg_fetch_array($result, NULL, PGSQL_ASSOC);
        list($header, $total) = each($totalArray);
        print "<hr><h3 style=\"text-align:left\"> Silver I Circle: $total </h3>";

        $query = " SELECT class, SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Silver I' group by class;";
        $result = pg_query ($query)
            or die ("Query failed");
        print "<table>\n";
        print "\t<tr>\n";
        print "\t\t<td>Class of</td>\n";
        print "\t\t<td>Amount Donated</td>\n";
        print "\t</tr>\n";
        while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
            print "\t<tr>\n";
            while(list($col_name, $col_value) = each($line)){
                print "\t\t<td>$col_value</td>\n";
            }
            print "\t</tr>\n";
        }
        print "</table>\n";

```

```

    $query = " SELECT DISTINCT name from circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Silver I';";
    $result = pg_query ($query)
        or die ("Query failed");
    print "<h3> Donors </h3>";
    print "<p>";
    while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
        while(list($col_name, $col_value) = each($line)){
            print "$col_value, ";
        }
    }
    print "</p>";

    #Silver II
    $query = " SELECT SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Silver II';";
    $result = pg_query ($query)
        or die ("Query failed");
    $totalArray = pg_fetch_array($result, NULL, PGSQL_ASSOC);
    list($header, $total) = each($totalArray);
    print "<hr><h3 style=\"text-align:left\"> Silver II Circle: $total </h3>";

    $query = " SELECT class, SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Silver II' group by class;";
    $result = pg_query ($query)
        or die ("Query failed");
    print "<table>\n";
    print "\t<tr>\n";
    print "\t\t<td>Class of</td>\n";

```

```

print "\t\t<td>Amount Donated</td>\n";
print "\t</tr>\n";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    print "\t<tr>\n";
    while(list($col_name, $col_value) = each($line)){
        print "\t\t<td>$col_value</td>\n";
    }
    print "\t</tr>\n";
}
print "</table>\n";

```

```

$query = " SELECT DISTINCT name from circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Silver II';";
$result = pg_query ($query)
    or die ("Query failed");
print "<h3> Donors </h3>";
print "<p>";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    while(list($col_name, $col_value) = each($line)){
        print "$col_value, ";
    }
}
print "</p>";

```

```

#Silver III
$query = " SELECT SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Silver III';";
$result = pg_query ($query)
    or die ("Query failed");
$totalArray = pg_fetch_array($result, NULL, PGSQL_ASSOC);

```

```

list($header, $total) = each($totalArray);
print "<hr><h3 style=\"text-align:left\"> Silver III Circle: $total </h3>";

$query = " SELECT class, SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Silver III' group by class;";
$result = pg_query ($query)
    or die ("Query failed");
print "<table>\n";
print "\t<tr>\n";
print "\t\t<td>Class of</td>\n";
print "\t\t<td>Amount Donated</td>\n";
print "\t</tr>\n";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    print "\t<tr>\n";
    while(list($col_name, $col_value) = each($line)){
        print "\t\t<td>$col_value</td>\n";
    }
    print "\t</tr>\n";
}
print "</table>\n";

$query = " SELECT DISTINCT name from circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Silver III';";
$result = pg_query ($query)
    or die ("Query failed");
print "<h3> Donors </h3>";
print "<p>";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    while(list($col_name, $col_value) = each($line)){
        print "$col_value, ";
    }
}

```



```

    }
}
print "</p>";

    #Bronze I
    $query = " SELECT SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Bronze I';";
    $result = pg_query ($query)
        or die ("Query failed");
    $totalArray = pg_fetch_array($result, NULL, PGSQL_ASSOC);
    list($header, $total) = each($totalArray);
    print "<hr><h3 style=\"text-align:left\"> Bronze I Circle: $total </h3>";

    $query = " SELECT class, SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Bronze I' group by class;";
    $result = pg_query ($query)
        or die ("Query failed");
    print "<table>\n";
    print "\t<tr>\n";
    print "\t\t<td>Class of</td>\n";
    print "\t\t<td>Amount Donated</td>\n";
    print "\t</tr>\n";
    while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
        print "\t<tr>\n";
        while(list($col_name, $col_value) = each($line)){
            print "\t\t<td>$col_value</td>\n";
        }
        print "\t</tr>\n";
    }
    print "</table>\n";

```

```

    $query = " SELECT DISTINCT name from circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Bronze I';";
    $result = pg_query ($query)
        or die ("Query failed");
    print "<h3> Donors </h3>";
    print "<p>";
    while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
        while(list($col_name, $col_value) = each($line)){
            print "$col_value, ";
        }
    }
    print "</p>";

    #Bronze II
    $query = " SELECT SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Bronze II';";
    $result = pg_query ($query)
        or die ("Query failed");
    $totalArray = pg_fetch_array($result, NULL, PGSQL_ASSOC);
    list($header, $total) = each($totalArray);
    print "<hr><h3 style=\"text-align:left\"> Bronze II Circle: $total </h3>";

    $query = " SELECT class, SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Bronze II' group by class;";
    $result = pg_query ($query)
        or die ("Query failed");
    print "<table>\n";
    print "\t<tr>\n";
    print "\t\t<td>Class of</td>\n";

```

```

print "\t\t<td>Amount Donated</td>\n";
print "\t</tr>\n";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    print "\t<tr>\n";
    while(list($col_name, $col_value) = each($line)){
        print "\t\t<td>$col_value</td>\n";
    }
    print "\t</tr>\n";
}
print "</table>\n";

```

```

$query = " SELECT DISTINCT name from circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Bronze II';";
$result = pg_query ($query)
    or die ("Query failed");
print "<h3> Donors </h3>";
print "<p>";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    while(list($col_name, $col_value) = each($line)){
        print "$col_value, ";
    }
}
print "</p>";

```

```

#Bronze III
$query = " SELECT SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Bronze III';";
$result = pg_query ($query)
    or die ("Query failed");
$totalArray = pg_fetch_array($result, NULL, PGSQL_ASSOC);

```

```

list($header, $total) = each($totalArray);
print "<hr><h3 style=\"text-align:left\"> Bronze III Circle: $total </h3>";

$query = " SELECT class, SUM(amount) FROM circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Bronze III' group by class;";
$result = pg_query ($query)
    or die ("Query failed");
print "<table>\n";
print "\t<tr>\n";
print "\t\t<td>Class of</td>\n";
print "\t\t<td>Amount Donated</td>\n";
print "\t</tr>\n";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    print "\t<tr>\n";
    while(list($col_name, $col_value) = each($line)){
        print "\t\t<td>$col_value</td>\n";
    }
    print "\t</tr>\n";
}
print "</table>\n";

$query = " SELECT DISTINCT name from circlenames WHERE date_part('year', date) =
date_part('year', CURRENT_DATE) AND circle = 'Bronze III';";
$result = pg_query ($query)
    or die ("Query failed");
print "<h3> Donors </h3>";
print "<p>";
while($line = pg_fetch_array($result, NULL, PGSQL_ASSOC)){
    while(list($col_name, $col_value) = each($line)){
        print "$col_value, ";
    }
}

```

```
        }  
    }  
    print "</p>";  
  
}
```

```
        pg_close($link);  
    ?>  
</body>  
  
</html>
```

donorUI.php

```
<html>

<head>
<meta name="GENERATOR" content="Microsoft FrontPage 5.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<title>Donor Table Access</title>
</head>

<style>
body{
    background-color: #1d2d50;
    color: white;
    text-align:center;
}

table{
    margin-top: 7px;
    margin-left: auto;
    margin-right: auto;
}

hr{
    color: white;
    background-color: white;
}

td{
```

```

padding: 5px;
}

</style>

<body>
<h1><center> Donor Table</center></h1>
<?php
    error_reporting(0);
    $link = pg_connect("host=itcsdbms user=rhodes dbname=sampleels")
        or die ("Could not connect to database pres");
    print ("Connected successfully\n");

    if ($_GET['Submit']) {

//      print("Input Called");
//Input of Data
    $fname = $_GET['firstname'];
    $fname = "'{$fname}'";
    $lname = $_GET['lastname'];
    $lname = "'{$lname}'";
    $phone = $_GET['phone'];
    $phone = "'{$phone}'";
    $email = $_GET['email'];
    if ($email == '') {$email = "NULL";}
    else $email = "'{$email}'";
    $street = $_GET['street'];
    $street = "'{$street}'";
    $grad = $_GET['grad'];
    if ($grad == '') {$grad = "NULL";}

```

```

else $grad = "'{$grad}'";
$donor = $_GET['donor'];
if ($donor == '') {$donor = "NULL";}
else $donor = "'{$donor}'";
$corp = $_GET['corp'];
if ($corp == '') {$corp = "NULL";}
else $corp = "'{$corp}'";
$zip = $_GET['zip'];
$zip = "'{$zip}'";
$spouse = $_GET['spouse'];
if ($spouse == '') {$spouse = "NULL";}
print "\t\t<td>${</td>\n";
$query = "insert into potentialdonor values (DEFAULT, $fname, $lname, $phone, $email,
$street, $grad, $donor, $corp, $zip, $spouse)";
$result = pg_query ($query)
    or die ("Input failed");
print "Data Entry Successful";

header("Location: http://jcsites.juniata.edu/students/sturtell9/databaseForm.html");
}
//Delete a row
else if ($_GET['Delete']) {
//print ("Delete called");

$donorId = $_GET['donorId'];

$query = "delete from potentialdonor where donorId = '$donorId'";
$result = pg_query ($query)
    or die ("Delete failed");
print "Deletion Successful";

```



```
        header("Location: http://jcsites.juniata.edu/students/sturtell9/databaseForm.html");  
    }  
  
    pg_close($link);  
    ?>  
</body>  
  
</html>
```

pledgeUI.php

```
<html>

<head>
<meta name="GENERATOR" content="Microsoft FrontPage 5.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<title>Donor Table Access</title>
</head>

<style>
body{
    background-color: #1d2d50;
    color: white;
    text-align:center;
}

table{
    margin-top: 7px;
    margin-left: auto;
    margin-right: auto;
}

hr{
    color: white;
    background-color: white;
}
```

```

td{
    padding: 5px;
}

</style>

<body>
<h1><center> Donor Table</center></h1>
<?php
    error_reporting(0);
    $link = pg_connect("host=itcsdbms user=rhodes dbname=sampleels")
        or die ("Could not connect to database pres");
    print ("Connected successfully\n");

    if ($_GET['Submit']) {

//      print("Input Called");
//Input of Data
        $pdate = $_GET['pledgeDate'];
        $pdate = "'{$pdate}'";
        $apdate = $_GET['antPayDate'];
        if ($apdate == '') {$apdate = "NULL";}
        else $apdate = "'{$apdate}'";
        $amount = $_GET['amountPledged'];
        $donorId = $_GET['donorId'];
        $projName = $_GET['projName'];
        if ($projName == '') {$projName = "NULL";}
        else $projName = "'{$projName}'";

        print "\t\t<td>${</td>\n";

```

```

        $query = "insert into pledge values (DEFAULT, $pdate, $apdate, $amount, $donorId,
$projName);";
        $result = pg_query ($query)
            or die ("Input failed");
        print "Data Entry Successful";

        header("Location: http://jcsites.juniata.edu/students/sturtell9/databaseForm.html");
    }
    //Delete a row
    else if ($_GET['Delete']) {
        //print ("Delete called");

        $pledgeId = $_GET['pledgeId'];

        $query = "delete from pledge where pledgeId = '$pledgeId';";
        $result = pg_query ($query)
            or die ("Delete failed");
        print "Deletion Successful";

        header("Location: http://jcsites.juniata.edu/students/sturtell9/databaseForm.html");
    }

    pg_close($link);
?>
</body>

</html>

```

donationUI.php

```
<html>

<head>
<meta name="GENERATOR" content="Microsoft FrontPage 5.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<title>Donor Table Access</title>
</head>

<style>
body{
    background-color: #1d2d50;
    color: white;
    text-align:center;
}

table{
    margin-top: 7px;
    margin-left: auto;
    margin-right: auto;
}

hr{
    color: white;
    background-color: white;
}

td{
```

```

padding: 5px;
}

</style>

<body>
<h1><center> Donor Table</center></h1>
<?php
    error_reporting(0);
    $link = pg_connect("host=itcsdbms user=rhodes dbname=sampleels")
        or die ("Could not connect to database pres");
    print ("Connected successfully\n");

    if ($_GET['Submit']) {

//      print("Input Called");
//Input of Data
    $pdate = $_GET['datePaid'];
    $pdate = "'{$pdate}'";
    $amount = $_GET['amountPaid'];
    $payMethod = $_GET['method'];
    if ($payMethod == '') {$payMethod = "NULL";}
    else $payMethod = "'{$payMethod}'";
    $pledgeId = $_GET['pledgeId'];

    print "\t\t<td>${</td>\n";
    $query = "insert into donation values ($pdate, $amount, $payMethod, $pledgeId)";
    $result = pg_query ($query)
        or die ("Input failed");
    print "Data Entry Successful";

```

```
    header("Location: http://jcsites.juniata.edu/students/sturtell9/databaseForm.html");
}
//Delete a row
else if ($_GET['Delete']) {
    //print ("Delete called");

    $donationId = $_GET['donationId'];

    $query = "delete from donation where donationId = '$donationId'";
    $result = pg_query ($query)
        or die ("Delete failed");
    print "Deletion Successful";

    header("Location: http://jcsites.juniata.edu/students/sturtell9/databaseForm.html");
}

pg_close($link);
?>
</body>

</html>
```

CSV Files

Project X

‘Science Building’, 100000000.00, 2025-05-01
 ‘Swimming Pool’, 500000.00, 2022-04-15
 ‘Movie Theater’, 123456.00, 2030-09-15

potentialDonor X

“Evan”, “Sturtevant”, “12089202550”, “sutrtel19@juniata.edu”, “718 East Ave”, “1998”, “ALUMNI”, “Google”, “16652”, null,
 ‘Loren’, ‘Rhodes’, ‘12345678901’, ‘rhodes@juniata.edu’, ‘123 Street St’, ‘2001’, ‘SENIOR’, ‘Juniata College’, ‘16652’, null,
 ‘Tyler’, ‘Paul’, ‘93875937594’, ‘paultj18@juniata.edu’, ‘345 Street st’, ‘1346’, ‘PARENTS’, null, ‘44454’, null,

pledge X

2019-12-25, 2020-01-25, 300.00, 1, Swimming Pool
 2020-06-15, 2020-07-15, 500000.00, 2, Movie Theater
 2019-11-16, 2019-12-16, 50.00, 3, Science Building

matchingCorporation X

Google, 987 Road Rd, 19786645129, google@gmail.com, 50.00, 9000000.00, 12345
 Juniata College, 1700 Moore St, 16875412355, juniata@juniata.edu, 35.50, 300000.00, 16652
 Walmart, 456 Walmart St, 12345432667, walmart@gmail.com, 70.00, 200.00, 16652

donation

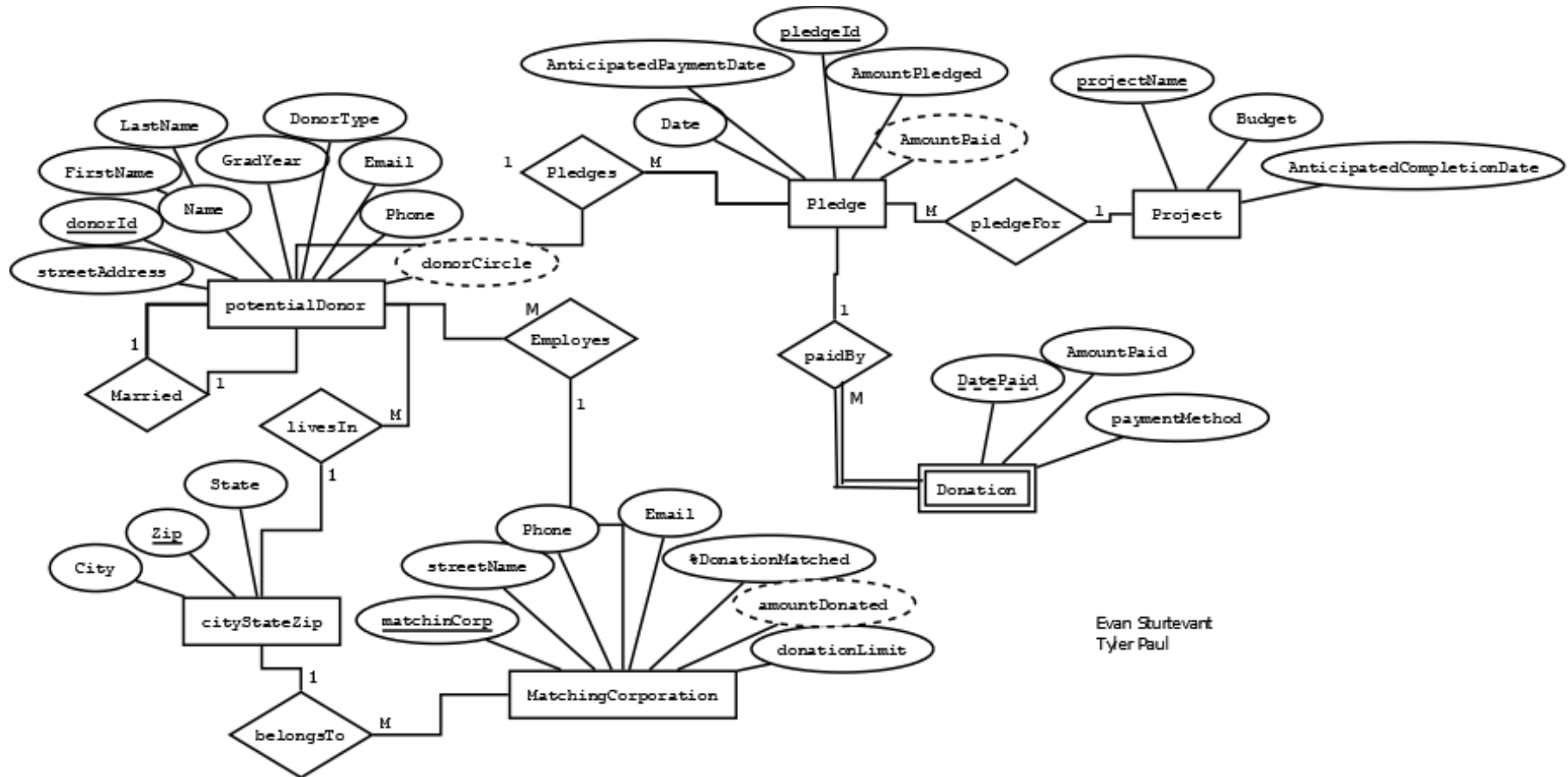
2020-01-12, 300.00, Check, 1
 2020-06-30, 100000.00, Credit Card, 2
 2019-12-03, 50.00, Cash, 3

cityStateZip X

68949, Holdrege, NE
 16652, Huntingdon, PA
 01864, North Reading, MA
 12345, Schenectady, NY

ARCHIVED VERSIONS BEYOND THIS PAGE

Current ER



Docs won't let us make a singular page landscape view, we decided to put the rest of the doc in landscape so that the ER dias could be formatted correctly.

DDL Revised

--Team: Evan Sturtevant, Tyler Paul

--Date 10/4/2020

DROP TABLE donation;

DROP TABLE pledge;

DROP TABLE potentialDonor;

DROP TABLE matchingCorporation;

DROP TABLE cityStateZip;

DROP TABLE project;

CREATE TABLE project

(projectName CHAR (20) NOT NULL UNIQUE,

budget NUMERIC (11,2),

compDate DATE,

PRIMARY KEY (projectName)

);

CREATE TABLE cityStateZip

(zip CHAR (5) NOT NULL UNIQUE,

city VARCHAR (20) NOT NULL,

state CHAR (2) NOT NULL,

PRIMARY KEY (zip)

);

CREATE TABLE matchingCorporation

(matchingCorp CHAR (20),
 streetAddress VARCHAR (50),
 phone CHAR (11),
 email VARCHAR (50),
 percentMatched NUMERIC (5,2),
 donationLimit NUMERIC (11,2),
 zip CHAR (5),

PRIMARY KEY (matchingCorp),

FOREIGN KEY (zip)

REFERENCES cityStateZip (zip)

);

CREATE TABLE potentialDonor

(donorId SERIAL,
 firstName VARCHAR (20) NOT NULL,
 lastName VARCHAR (20) NOT NULL,
 phone CHAR (11) NOT NULL,
 email VARCHAR (50),
 streetAddress VARCHAR (50) NOT NULL,
 gradYear CHAR (4),

```

donorType      CHAR    (10)  NOT NULL,
matchingCorp   CHAR    (20),
zip            CHAR    (5)    NOT NULL,
spouseId       INTEGER,

```

```

PRIMARY KEY (donorId),

```

```

FOREIGN KEY (matchingCorp)

```

```

    REFERENCES matchingCorporation (matchingCorp),

```

```

FOREIGN KEY (zip)

```

```

    REFERENCES cityStateZip (zip),

```

```

FOREIGN KEY (spouseId)

```

```

    REFERENCES potentialDonor (donorId),

```

```

CHECK (donorType IN ('SENIOR', 'ALUMNI', 'PARENTS', 'FACULTY', 'ADMINISTRATOR', 'STAFF', 'CORPORATIONS',
'OTHER'))

```

```

);

```

```

CREATE TABLE pledge

```

```

(pledgeId      SERIAL,
date           DATE          NOT NULL,
anticipatedPaymentDate DATE,
amountPledged  NUMERIC (11,2) NOT NULL,
donorId        INTEGER       NOT NULL,
projectName    char    (20),

```

```

PRIMARY KEY (pledgeId),

```

```

FOREIGN KEY (donorId)

```

```

    REFERENCES potentialDonor (donorID),

```

```
FOREIGN KEY (projectName)
  REFERENCES project (projectName)
);
```

```
CREATE TABLE donation
(datePaid      DATE      NOT NULL,
amountPaid    NUMERIC (8,2) NOT NULL,
paymentMethod CHAR  (6)   NOT NULL,
pledgeId      INTEGER    NOT NULL,
```

```
PRIMARY KEY (datePaid, pledgeId),
FOREIGN KEY (pledgeId)
  REFERENCES pledge (pledgeId)
);
```

Script Running

Script started on October 14, 2020 07:24:23 PM UTC

Finished my .bashrc

sturtel19@Solaris:~\$ psql sampleels

psql (9.6.1)

Type "help" for help.

sampleels=> \d [Ki databaseDDL.sql

DROP TABLE

DROP TABLE

DROP TABLE

DROP TABLE

DROP TABLE

DROP TABLE

CREATE TABLE

CREATE TABLE

CREATE TABLE

CREATE TABLE

CREATE TABLE

CREATE TABLE

sampleels=> ^D\q

sturtel19@Solaris:~\$

script done on October 14, 2020 07:24:41 PM UTC

CSV Files

Project X

‘Science Building’, 100000000.00, 2025-05-01
 ‘Swimming Pool’, 500000.00, 2022-04-15
 ‘Movie Theater’, 123456.00, 2030-09-15

potentialDonor X

“Evan”, “Sturtevant”, “12089202550”, “sutrtel19@juniata.edu”, “718 East Ave”, “1998”, “ALUMNI”, “Google”, “16652”, null,
 ‘Loren’, ‘Rhodes’, ‘12345678901’, ‘rhodes@juniata.edu’, ‘123 Street St’, ‘2001’, ‘SENIOR’, ‘Juniata College’, ‘16652’, null,
 ‘Tyler’, ‘Paul’, ‘93875937594’, ‘paultj18@juniata.edu’, ‘345 Street st’, ‘1346’, ‘PARENTS’, null, ‘44454’, null,

pledge X

2019-12-25, 2020-01-25, 300.00, 1, Swimming Pool
 2020-06-15, 2020-07-15, 500000.00, 2, Movie Theater
 2019-11-16, 2019-12-16, 50.00, 3, Science Building

matchingCorporation X

Google, 987 Road Rd, 19786645129, google@gmail.com, 50.00, 9000000.00, 12345
 Juniata College, 1700 Moore St, 16875412355, juniata@juniata.edu, 35.50, 300000.00, 16652
 Walmart, 456 Walmart St, 12345432667, walmart@gmail.com, 70.00, 200.00, 16652

donation

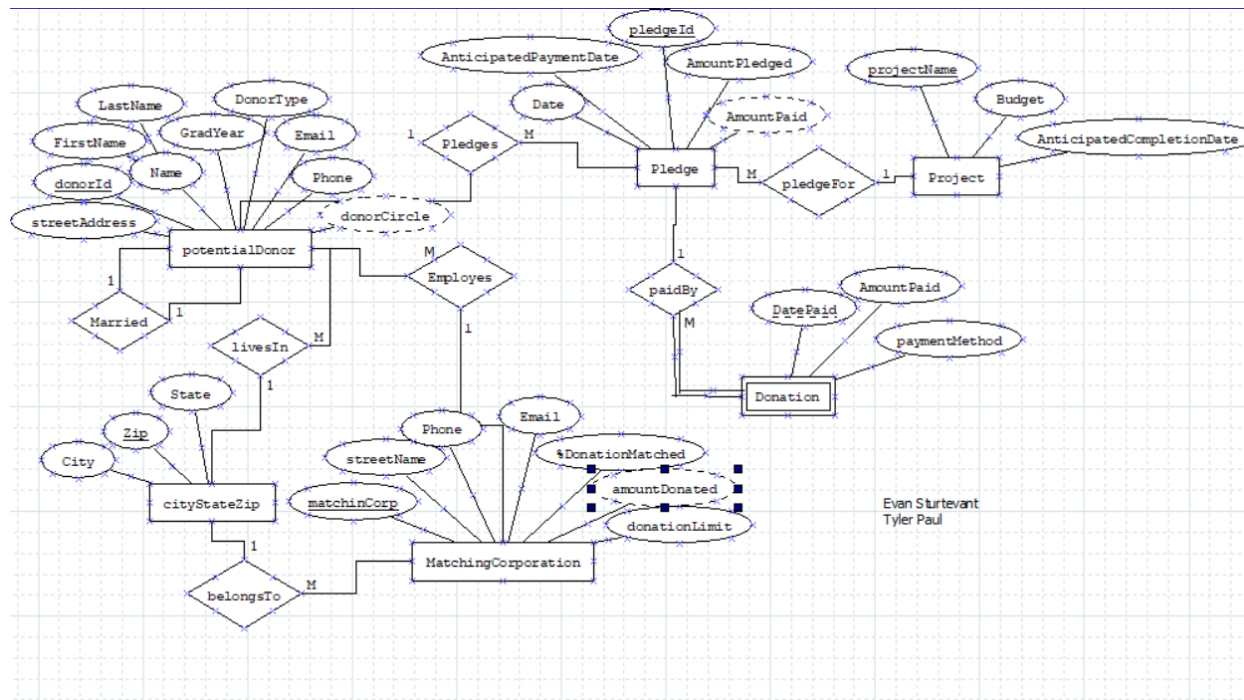
2020-01-12, 300.00, Check, 1
 2020-06-30, 100000.00, Credit Card, 2
 2019-12-03, 50.00, Cash, 3

cityStateZip X

68949, Holdrege, NE
 16652, Huntingdon, PA
 01864, North Reading, MA
 12345, Schenectady, NY

44454, Petersburg, OH

ER Dia 3



SQL Script

--Team: Evan Sturtevant, Tyler Paul

--Date 10/4/2020

DROP TABLE donation;

DROP TABLE pledge;

DROP TABLE potentialDonor;

DROP TABLE matchingCorp;

DROP TABLE cityStateZip;

DROP TABLE project;

CREATE TABLE project

(projectName CHAR (20),

budget NUMERIC (11,2),

compDate DATE,

CONSTRAINT project_projectName_pk PRIMARY KEY (projectName)

);

CREATE TABLE cityStateZip

(zip CHAR (5),

city VARCHAR (20) NOT NULL,

state char (2) NOT NULL,

CONSTRAINT csz_zip_pk PRIMARY KEY (zip)

);

```

CREATE TABLE matchingCorporation
(matchingCorp CHAR (20),
streetAddress VARCHAR (50),
phone CHAR (11),
email VARCHAR (50),
percentMatched NUMERIC (5,2),
donationLimit NUMERIC (8,2),
zip CHAR (5),
CONSTRAINT corporation_matchingCorp_pk PRIMARY KEY (matchingCorp),
CONSTRAINT corporation_zip_fk FOREIGN KEY (zip) REFERENCES cityStateZip (zip)
);

```

```

CREATE TABLE potentialDonor
(donorId SERIAL,
firstName VARCHAR (20),
lastName VARCHAR (20),
phone CHAR (11),
email VARCHAR (50),
streetAddress VARCHAR(50),
gradYear CHAR(4),
donorType CHAR (10),
matchingCorp CHAR (20),
zip CHAR (5),
spouseId INTEGER,
CONSTRAINT pDonor_donorId_pk PRIMARY KEY (donorId),
CONSTRAINT pDonor_matchingCorp_fk FOREIGN KEY (matchingCorp) REFERENCES matchingCorporation (matchingCorp),
CONSTRAINT pDonor_zip_fk FOREIGN KEY (zip) REFERENCES cityStateZip (zip),
CONSTRAINT pDonor_spouseId_fk FOREIGN KEY (spouseId) REFERENCES potentialDonor (donorId),

```

```

CONSTRAINT pDonor_donorType_check CHECK (donorType IN ('SENIOR', 'ALUMNI', 'PARENTS', 'FACULTY', 'ADMINISTRATOR',
'STAFF', 'CORPORATIONS', 'OTHER'))
);

```

```

CREATE TABLE pledge
(pledgeId SERIAL,
date DATE,
anticipatedPaymentDate DATE,
amountPledged NUMERIC (8,2),
donorId INTEGER,
projectName char (20),
CONSTRAINT pledge_pledgeId_pk PRIMARY KEY (pledgeId),
CONSTRAINT pledge_donorId_fk FOREIGN KEY (donorId) REFERENCES potentialDonor (donorID),
CONSTRAINT pledge_projectName_fk FOREIGN KEY (projectName) REFERENCES project (projectName)
);

```

```

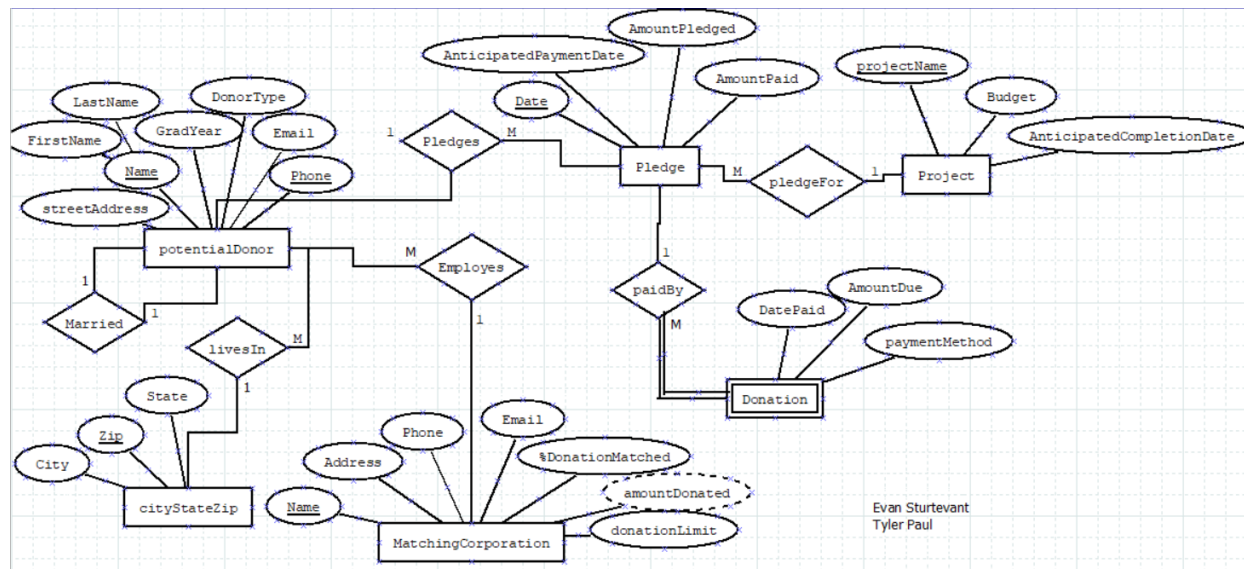
CREATE TABLE donation
(datePaid DATE,
amountPaid NUMERIC (8,2) NOT NULL,
paymentMethod CHAR (6),
pledgeId INTEGER NOT NULL,
CONSTRAINT donation_datePaid_pk PRIMARY KEY (datePaid, pledgeId),
CONSTRAINT donation_pledgeId_fk FOREIGN KEY (pledgeId) REFERENCES pledge (pledgeId)
);

```

Relational Schema

- potentialDonor(donorId, firstName, lastName, phone, email, streetAddress, gradYear, donorType, *matchingCorp*, *zip*, *spouseId*)
- pledge(pledgeId, date, anticipatedPaymentDate, amountPledged, *donorId*, *projectName*)
- project(projectName, budget, compDate)
- donation(datePaid, amountPaid, paymentMethod, *pledgeId*)
- matchingCorporation(matchingCorp, streetName, phone, email, percentMatched, donationLimit, *zip*)
- cityStateZip(zip, state, city)

ER Draft 2:



ER Draft 1:

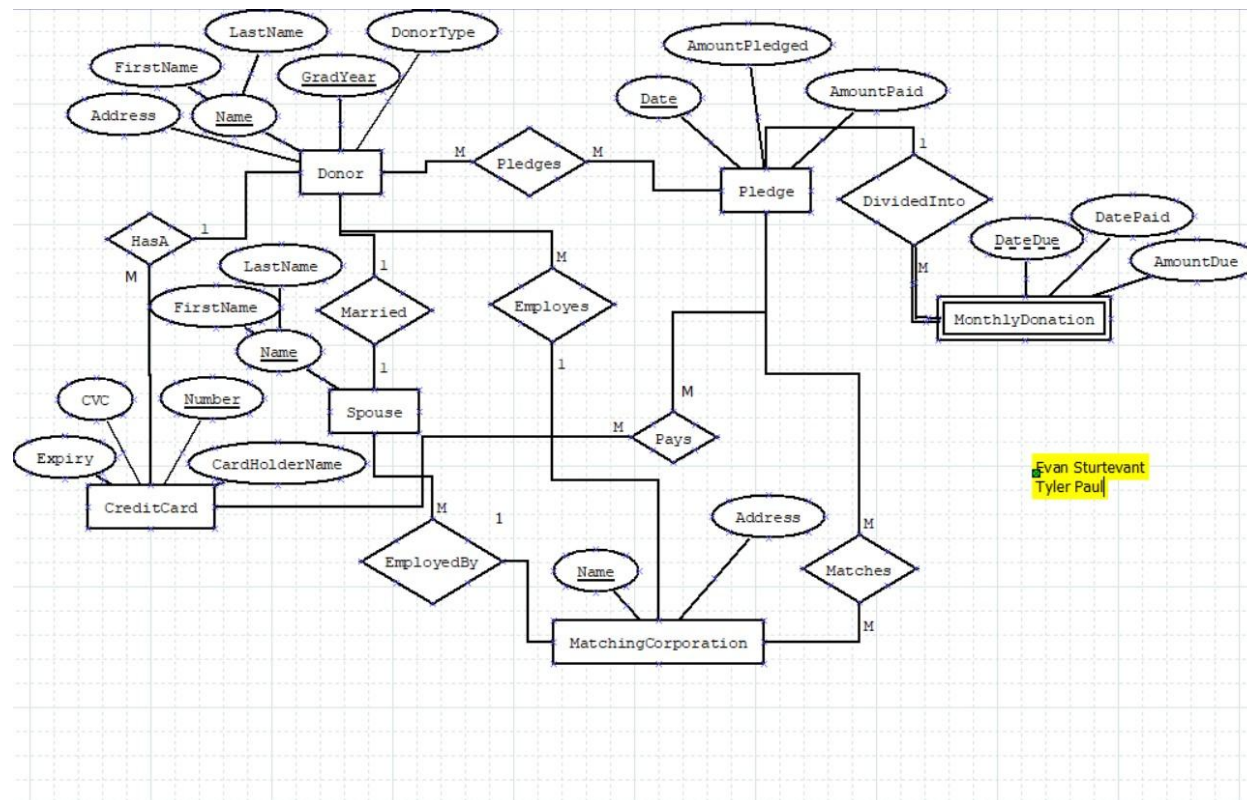


Table of Contents

1. Overview

- Description.....3
- Constraints.....3

1.1 Forms

- Annual Fund Gift.....4

1.2 Reports

- Annual Report to Donors.....7
- Payments Due Report.....10
- Monthly Report.....11
- Classic Representative Contact List.....12
- Phonathon Volunteer Contact List.....13

1. Overview

General Overview

For the course Database Management Systems, our team consisting of Tyler Paul, Evan Sturtevant, and Brenden Price will be creating a database for the Beta University Annual Fund. With a wide range of potential donors, it is imperative that Suzanne Hayes has a database developed to manage the responsibilities of both raising funds and keeping track of donations. A number of different forms and reports will be generated from the database to help the university distinguish the donations made amongst the potential donors.

Constraints

Deadline: The deadline for the Beta University Annual Fund database will be November 20-23, 2020, tentatively (based on presentation date).

Covid-19: At this point in time, the project won't have any constraints pertaining to the resources at our disposal. The biggest constraint we face as a group is the uncertainty of the pandemic. Issues could potentially arise if we were to switch to strictly online classes and/or be sent home. As a group, we will address these issues in the event that either of them were to occur.

Beta University Annual Donation Form:

Description:

This form is filled out by people who wish to donate to the University. The donator will fill out the entire form, leaving anything non-applicable blank (i.e. credit card info for a check payment).

Info input by the user:

First Name, Last Name

- Simple text box

Type of Donor

- Drop Down menu
 - Alumni, Student, Parent, Administrator

Today's Date

- mm/dd/yyyy

Graduation Year

4-digit Year

Amount Pledged, Amount Enclosed

- \$ Amount

of Payments

- Number of monthly payments; number 1-12

Payment Type

- Drop Down Menu
 - Credit, Check

Cardholder Name

- Simple text box

Card Number

- 16 Digits

Expiry

- mm/yyyy

CVC

- 3 Digits

Matching Corporation, Street Address, Street Address 2, City, State/Province

- Simple text box

Zip Code

- 5 Digits

Select Country





- Drop Down Menu

Name of Spouse

- Simple text box

Beta University Annual Donation Form

First Name	Type of Donor ▼
Last Name	Graduation Year
Today's Date	

Amount Pledged	Amount Enclosed	# of Payments ▼
Payment Type ▼		
 Cardholder's Name		
 Card Number		
Expiry		
MM/YYYY		 CVC
🔒 128-bit secured		

Matching Corporation Name	
Street Address (of Corporation)	
Street Address 2	
City	State / Province
Postal / Zip Code	Select Country ▼
Name of Spouse (if matching cooperation is spouse's employer)	

Annual Report to Donors:

Description: This 2 page report is mailed to donors each year. It details the yearly earnings from donations, and it credits many of the donors. It includes many statistics about that year's donations and the different groups that donated.

Annual Report

President's Circle

Total Donated:
20,500,000

Jewell Juel, Alexia Takacs ,Ignacia Dale ,Susana Mcmurtry ,Aiko Wethington ,Stevie Greenburg ,Francisca Tancredi ,Lacy Alper ,Argelia Voth ,Rosie Slovak ,Vickey Pell ,Ozella Coblenitz ,Fausto Babineaux ,Cierra Slee ,Lady Higginbottom ,Loni Nolin ,Toby Delpino ,Yen Bullion ,Janell Quiroz ,Cristobal Sweetland ,Cathleen Vaugh ,Ramona Holst ,Brienne Raborn ,Elwood Etchison ,Solomon Callan ,Brittanie Fearon ,Lanny Izzo ,Patrina Stickie ,Otis Yamasaki ,Latoya Huhn ,Nelida Savory ,Takisha Hebel ,Mark Bossi ,Hilaria Sue ,Milagros Siu ,Mario Profit ,Clorinda Delillo ,Keri Dupre ,Chanel Zank ,Carmina Lando ,Elouise Gravely ,Jamel Bengston ,Elissa Letellier ,Blanche Raposo ,Kandace Claire ,Stasia Chaves ,Briana Peloquin ,Haydee Kurz ,Gerald Tabares ,Jeannette Morey

Platinum Circle

Total Donated:
16,000,000

Susana Mcmurtry ,Aiko Wethington ,Stevie Greenburg ,Francisca Tancredi ,Lacy Alper ,Argelia Voth ,Rosie Slovak ,Vickey Pell ,Ozella Coblenitz ,Fausto Babineaux ,Cierra Slee ,Lady Higginbottom ,Loni Nolin ,Toby Delpino ,Yen Bullion ,Janell Quiroz ,Cristobal Sweetland ,Cathleen Vaugh ,Ramona Holst ,Brienne Raborn ,Elwood Etchison ,Solomon Callan ,Brittanie Fearon ,Lanny Izzo ,Patrina Stickie ,Otis Yamasaki ,Latoya Huhn ,Nelida Savory ,Takisha Hebel ,Mark Bossi ,Hilaria Sue ,Milagros Siu ,Mario Profit ,Clorinda Delillo ,Keri Dupre ,Chanel Zank ,Carmina Lando ,Elouise Gravely ,Jamel Bengston ,Elissa Letellier ,Blanche Raposo ,Kandace Claire ,Stasia Chaves ,Briana Peloquin ,Haydee Kurz ,Gerald Tabares ,Jeannette Morey

Gold I Circle

Total Donated:
12,000,000

Jewell Juel, Alexia Takacs ,Ignacia Dale ,Susana Mcmurtry ,Aiko Wethington ,Stevie Greenburg ,Francisca Tancredi ,Lacy Alper ,Argelia Voth ,Rosie Slovak ,Vickey Pell ,Ozella Coblenitz ,Fausto Babineaux ,Cierra Slee ,Lady Higginbottom ,Loni Nolin ,Toby Delpino ,Yen Bullion ,Janell Quiroz ,Cristobal Sweetland ,Cathleen Vaugh ,Ramona Holst ,Brienne Raborn ,Elwood Etchison ,Solomon Callan ,Brittanie Fearon ,Lanny Izzo ,Patrina Stickie ,Otis Yamasaki ,Latoya Huhn ,Nelida Savory ,Takisha Hebel ,Mark Bossi ,Hilaria Sue ,Milagros Siu ,Mario Profit ,Clorinda Delillo ,Keri Dupre ,Chanel Zank ,Carmina Lando ,Elouise Gravely ,Jamel Bengston ,Elissa Letellier ,Blanche Raposo ,Kandace Claire ,Stasia Chaves ,Briana Peloquin ,Haydee Kurz ,Gerald Tabares ,Jeannette Morey ,Ramona Holst ,Brienne Raborn ,Elwood Etchison ,Solomon Callan ,Brittanie Fearon ,Lanny Izzo ,Patrina Stickie ,Otis Yamasaki ,Latoya Huhn

Gold II Circle

Total Donated:
11,000,000

Jewell Juel, Alexia Takacs ,Ignacia Dale ,Susana Mcmurtry ,Aiko Wethington ,Stevie Greenburg ,Francisca Tancredi ,Lacy Alper ,Argelia Voth ,Rosie Slovak ,Vickey Pell ,Ozella Coblenitz ,Fausto Babineaux ,Cierra Slee ,Lady Higginbottom ,Loni Nolin ,Lanny Izzo ,Patrina Stickie ,Otis Yamasaki ,Latoya Huhn ,Nelida Savory ,Takisha Hebel ,Mark Bossi ,Hilaria Sue ,Milagros Siu ,Mario Profit ,Clorinda Delillo ,Keri Dupre ,Chanel Zank ,Carmina Lando ,Elouise Gravely ,Jamel Bengston ,Elissa Letellier ,Blanche Raposo ,Kandace Claire ,Stasia Chaves ,Briana Peloquin ,Haydee Kurz ,Gerald Tabares ,Jeannette Morey

Silver I Circle

Total Donated:
10,000,000

Jewell Juel, Alexia Takacs ,Ignacia Dale ,Susana Mcmurtry ,Aiko Wethington ,Stevie Greenburg ,Francisca Tancredi ,Lacy Alper ,Argelia Voth ,Rosie Slovak ,Vickey Pell ,Ozella Coblenitz ,Fausto Babineaux ,Cierra Slee ,Lady Higginbottom ,Loni Nolin ,Toby Delpino ,Yen Bullion ,Janell Quiroz ,Cristobal Sweetland ,Cathleen Vaugh ,Ramona Holst ,Brienne Raborn ,Elwood Etchison ,Solomon Callan ,Brittanie Fearon ,Lanny Izzo ,Patrina Stickie ,Otis Yamasaki ,Latoya Huhn ,Nelida Savory ,Takisha Hebel ,Mark Bossi ,Hilaria Sue ,Milagros Siu ,Mario Profit ,Clorinda Delillo ,Keri Dupre ,Chanel Zank ,Carmina Lando ,Elouise Gravely ,Jamel Bengston ,Elissa Letellier ,Blanche Raposo ,Kandace Claire ,Stasia Chaves ,Briana Peloquin ,Haydee Kurz ,Gerald Tabares ,Jeannette Morey

Silver II Circle

Total Donated:
900,000

Susana Mcmurtry ,Aiko Wethington ,Stevie Greenburg ,Francisca Tancredi ,Lacy Alper ,Argelia Voth ,Rosie Slovak ,Vickey Pell ,Ozella Coblenitz ,Fausto Babineaux ,Cierra Slee ,Lady Higginbottom ,Loni Nolin ,Toby Delpino ,Yen Bullion ,Janell Quiroz ,Cristobal Sweetland ,Cathleen Vaugh ,Ramona Holst ,Brienne Raborn ,Elwood Etchison ,Solomon Callan ,Brittanie Fearon ,Lanny Izzo ,Patrina Stickie ,Otis Yamasaki ,Latoya Huhn ,Nelida Savory ,Takisha Hebel ,Mark Bossi ,Hilaria Sue ,Milagros Siu ,Mario Profit ,Clorinda Delillo ,Keri Dupre ,Chanel Zank ,Carmina Lando ,Elouise Gravely ,Jamel Bengston ,Elissa Letellier ,Blanche Raposo ,Kandace Claire ,Stasia Chaves ,Briana Peloquin ,Haydee Kurz ,Gerald Tabares ,Jeannette Morey

Silver III Circle

Total Donated:
850,000

Jewell Juel, Alexia Takacs ,Ignacia Dale ,Susana Mcmurtry ,Aiko Wethington ,Stevie Greenburg ,Francisca Tancredi ,Lacy Alper ,Argelia Voth ,Rosie Slovak ,Vickey Pell ,Ozella Coblenitz ,Fausto Babineaux ,Cierra Slee ,Lady Higginbottom ,Loni Nolin ,Toby Delpino ,Yen Bullion ,Janell Quiroz ,Cristobal Sweetland ,Cathleen Vaugh ,Ramona Holst ,Brienne Raborn ,Elwood Etchison ,Solomon Callan ,Brittanie Fearon ,Lanny Izzo ,Patrina Stickie ,Otis Yamasaki ,Latoya Huhn ,Nelida Savory ,Takisha Hebel ,Mark Bossi ,Hilaria Sue ,Milagros Siu ,Mario Profit ,Clorinda Delillo ,Keri Dupre ,Chanel Zank ,Carmina Lando ,Elouise Gravely ,Jamel Bengston ,Elissa Letellier ,Blanche Raposo ,Kandace Claire ,Stasia Chaves ,Briana Peloquin ,Haydee Kurz ,Gerald Tabares ,Jeannette Morey ,Ramona Holst ,Brienne Raborn ,Elwood Etchison ,Solomon Callan ,Brittanie Fearon ,Lanny Izzo ,Patrina Stickie ,Otis Yamasaki ,Latoya Huhn

Bronze I Circle

Total Donated:
750,000

Jewell Juel, Alexia Takacs ,Ignacia Dale ,Susana Mcmurtry ,Aiko Wethington ,Stevie Greenburg ,Francisca Tancredi ,Lacy Alper ,Argelia Voth ,Rosie Slovak ,Vickey Pell ,Ozella Coblenitz ,Fausto Babineaux ,Cierra Slee ,Lady Higginbottom ,Loni Nolin ,Lanny Izzo ,Patrina Stickie ,Otis Yamasaki ,Latoya Huhn ,Nelida Savory ,Takisha Hebel ,Mark Bossi ,Hilaria Sue ,Milagros Siu ,Mario Profit ,Clorinda Delillo ,Keri Dupre ,Chanel Zank ,Carmina Lando ,Elouise Gravely ,Jamel Bengston ,Elissa Letellier ,Blanche Raposo ,Kandace Claire ,Stasia Chaves ,Briana Peloquin ,Haydee Kurz ,Gerald Tabares ,Jeannette Morey

Bronze II Circle

Total Donated:
500,000

Jewell Juel, Alexia Takacs ,Ignacia Dale ,Susana Mcmurtry ,Aiko Wethington ,Stevie Greenburg ,Francisca Tancredi ,Lacy Alper ,Argelia Voth ,Rosie Slovak ,Vickey Pell ,Ozella Coblenitz ,Fausto Babineaux ,Cierra Slee ,Lady Higginbottom ,Loni Nolin ,Toby Delpino ,Yen Bullion ,Janell Quiroz ,Cristobal Sweetland ,Cathleen Vaugh ,Ramona Holst ,Brienne Raborn ,Elwood Etchison ,Solomon Callan ,Brittanie Fearon ,Lanny Izzo ,Patrina Stickie ,Otis Yamasaki ,Latoya Huhn ,Nelida Savory ,Takisha Hebel ,Mark Bossi ,Hilaria Sue ,Milagros Siu ,Mario Profit ,Clorinda Delillo ,Keri Dupre ,Chanel Zank ,Carmina Lando ,Elouise Gravely ,Jamel Bengston ,Elissa Letellier ,Blanche Raposo ,Kandace Claire ,Stasia Chaves ,Briana Peloquin ,Haydee Kurz ,Gerald Tabares ,Jeannette Morey ,Ramona Holst ,Brienne Raborn ,Elwood Etchison ,Solomon Callan ,Brittanie Fearon ,Lanny Izzo ,Patrina Stickie ,Otis Yamasaki ,Latoya Huhn

Bronze III Circle

Total Donated:
450,000

Jewell Juel, Alexia Takacs ,Ignacia Dale ,Susana Mcmurtry ,Aiko Wethington ,Stevie Greenburg ,Francisca Tancredi ,Lacy Alper ,Argelia Voth ,Rosie Slovak ,Vickey Pell ,Ozella Coblenitz ,Fausto Babineaux ,Cierra Slee ,Lady Higginbottom ,Loni Nolin ,Lanny Izzo ,Patrina Stickie ,Otis Yamasaki ,Latoya Huhn ,Nelida Savory ,Takisha Hebel ,Mark Bossi ,Hilaria Sue ,Milagros Siu ,Mario Profit ,Clorinda Delillo ,Keri Dupre ,Chanel Zank ,Carmina Lando ,Elouise Gravely ,Jamel Bengston ,Elissa Letellier ,Blanche Raposo ,Kandace Claire ,Stasia Chaves ,Briana Peloquin ,Haydee Kurz ,Gerald Tabares ,Jeannette Morey

Year	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
Amount in \$	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	
% Participation	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	50%	

Total Amount: \$1,234,567,890

Payments Due Report:

Payments Due

Name	Amount Due	Date Due	Address	Pledge	Amount Paid	Previous Payment Date
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020
Example Output	\$57	10/15/2020	123 House Street , TownVile PA 01234	\$300	\$243	9/15/2020

Monthly Report:

Description:

Report intended for internal use. This report must display the total current amount that has been pledged, the current total amount received of the current total pledged, and calculate and display the percentage received of the total.

Overall Totals		
Total Amount Pledged	Amount Received of Total Pledged	Percentage of Pledges Received
150000000	100000000	66%
President's		
Total Amount Pledged	Amount Received of Total Pledged	Percentage of Pledges Received
150000000	100000000	66%
Platinum		
Total Amount Pledged	Amount Received of Total Pledged	Percentage of Pledges Received
150000000	100000000	66%
Gold I		
Total Amount Pledged	Amount Received of Total Pledged	Percentage of Pledges Received
150000000	100000000	66%
Gold II		
Total Amount Pledged	Amount Received of Total Pledged	Percentage of Pledges Received
150000000	100000000	66%
Silver I		
Total Amount Pledged	Amount Received of Total Pledged	Percentage of Pledges Received
150000000	100000000	66%
Silver II		
Total Amount Pledged	Amount Received of Total Pledged	Percentage of Pledges Received
150000000	100000000	66%
Silver III		
Total Amount Pledged	Amount Received of Total Pledged	Percentage of Pledges Received
150000000	100000000	66%
Bronze I		
Total Amount Pledged	Amount Received of Total Pledged	Percentage of Pledges Received
150000000	100000000	66%
Bronze II		
Total Amount Pledged	Amount Received of Total Pledged	Percentage of Pledges Received
150000000	100000000	66%
Bronze III		
Total Amount Pledged	Amount Received of Total Pledged	Percentage of Pledges Received
150000000	100000000	66%

This report is generated monthly. It is generated for the purpose of mailing reminders to pledgers that are late on their monthly payment.

Class Representative Contact List <Report>:

Description:

_____ This will be an internal report describing pledge contacts for a class representative, providing names, addresses, phone numbers, donation information from previous year, and donation information for the current year. Intended use is to provide the class representative with information to contact more potential donators.

John Smith's Contact's				
Name:	Address:	Phone #:	Previous Year Donations	Current Year Donations
Abe Dunn	125 Candy Cane Ln, Westfield, Georgia, 111111	(123)456-7890	\$100	\$75
Smith Wesley	127 Candy Cane Ln, Westfield, Georgia, 111111	(234)567-8901	\$2,500	\$1,500
George Dellini	129Candy Cane Ln, Westfield, Georgia, 111111	(345)678-9012	\$400	\$700
Jeff Bond	131 Candy Cane Ln, Westfield, Georgia, 111111	(456)789-0123	\$125	\$150

Phonathon Volunteer Contact List <Report>:

Description:

This report will generate the contact list of donors that the volunteer will contact concerning donations. The report includes attributes such as Name, Phone #, Address, Category (type of donor), Graduation Year, and Last Year Donation Information. The form below can be utilized when a new donor is registered and added to the database.

Phonathon Volunteer Contact List

Name	
Street Address	
City	State / Province
Postal / Zip Code	Select Country ▼
Select Category ▼	
Graduation Year	
Last Year Donor Information	

John Smith Contact List					
Name:	Address:	Phone #:	Category:	Graduation Year:	Last Year Donor Information:
Griffin Davie	123 House Rd. Blue, NC 12345	(012) 345-6789	Alumnus	1993	Donated \$50
Terrence Mayer	345 Right Ln. Huntingdon, PA 16652	(234) 987-3456	Alumnus	2001	Donated \$250
Holly Lott	432 Red Rd. Ladder, TX 98765	(432) 345-0101	Alumna	1986	Donated \$100
Kara Wolf	762 Main St. Green, GA 83211	(888) 402-1010	Alumna	1989	Donated \$25
Beverly Burris	543 Train Ave. North, NY 12445	(876) 654-0955	Administrator	2005	Donated \$500