Atlanta zoon

­

Ruiyang Qin

drop database if exists zoo;

create database zoo;

use zoo;

## Building ---------------------------------------------------------------------------------

drop table IF EXISTS zoo\_user;

create table zoo\_user (

email varchar(40) NOT NULL UNIQUE ,

username varchar(40) NOT NULL UNIQUE,

ppassword varchar(40) not null,

usertype enum('admin', 'staff', 'visitor') default 'visitor' not null,

primary key (email)

)ENGINE=INNODB;

drop table IF EXISTS admin;

create table admin (

email varchar(40),

username VARCHAR(40),

PRIMARY KEY (email),

foreign key (email) references zoo\_user(email) ON UPDATE CASCADE ON DELETE CASCADE

)ENGINE=INNODB;

drop table IF EXISTS staff;

create table staff (

email varchar(40),

username VARCHAR(40),

PRIMARY KEY (email),

foreign key (email) references zoo\_user(email) ON UPDATE CASCADE on DELETE CASCADE

)ENGINE=INNODB;

drop table IF EXISTS visitor;

create table visitor (

email varchar(40),

username VARCHAR(40),

PRIMARY KEY (email),

foreign key (email) references zoo\_user(email) ON UPDATE CASCADE ON DELETE CASCADE

)ENGINE=INNODB;

drop table IF EXISTS exhibit;

create table exhibit (

nname varchar(40), # not null unique

waterfeature boolean not null,

size int(11) not null,

primary key (nname)

)ENGINE=INNODB;

drop table IF EXISTS animal;

create table animal (

nname varchar(40) ,

species varchar(40),

animal\_type ENUM('Mammal', 'Bird', 'Amphibian', 'Reptile', 'Fish', 'Invertebrate') not null,

age int not null,

animalexhibit varchar(40),

primary key (nname, species),

foreign key (animalexhibit) references exhibit(nname) on update cascade on delete restrict

)ENGINE=INNODB;

drop table IF EXISTS zoo\_show;

create table zoo\_show (

nname varchar(40),

dateandtime datetime,

showexhibit varchar(40),

showstaff varchar(40),

showdate DATE,

primary key (nname, dateandtime),

foreign key (showexhibit) references exhibit(nname) on update cascade on delete RESTRICT,

foreign key (showstaff) references staff(email) on update cascade on delete cascade

)ENGINE=INNODB;

drop table IF EXISTS showvisits;

create table showvisits (

email varchar (40),

nname varchar(40),

dateandtime datetime,

primary key (email, nname, dateandtime),

foreign key (email) references visitor(email) on update cascade on delete cascade,

foreign key (nname, dateandtime) references zoo\_show(nname,dateandtime) on update cascade on delete cascade

)ENGINE=INNODB;

drop table IF EXISTS exhibitvisits;

create table exhibitvisits (

email varchar(40),

nname varchar(40),

dateandtime datetime,

primary key (email, nname, dateandtime),

foreign key (email) references visitor(email) on update cascade on delete cascade,

foreign key (nname) references exhibit(nname) on update cascade on delete restrict

)ENGINE=INNODB;

drop table IF EXISTS note;

create table note (

nname varchar(40),

species varchar(40),

email varchar (40),

notetime datetime not null,

notetext text,

primary key (nname, species, email, notetime),

foreign key (nname, species) references animal(nname, species) on update cascade on delete cascade,

foreign key (email) references staff(email) on update cascade on delete cascade

)ENGINE=INNODB;

### ======================================================================================================

### functionality-----------------------------------------------------------------------------------------

## login and register=================================================================================

SELECT email, username FROM zoo\_user where email = '' and username = '';

INSERT INTO zoo\_user(email, username, ppassword, usertype) VALUES ('','','','','');

INSERT INTO admin(email, username) VALUES ('','');

INSERT INTO visitor(email, username) VALUES ('','');

INSERT INTO staff(email, username) VALUES ('','');

SELECT username, ppassword, usertype FROM zoo\_user WHERE username='' AND ppassword=MD5('');

## sort===============================================================================================

# animal sort

SELECT nname, species, animalexhibit, age, animal\_type from animal where nname = 'add any constraints here' ORDER BY nname ASC;

SELECT nname, species, animalexhibit, age, animal\_type from animal where nname = 'add any constraints here' ORDER BY nname DESC;

# show sort

SELECT nname, showexhibit, dateandtime from zoo\_show where nname = 'add any constraints here' ORDER BY nname ASC;

SELECT nname, showexhibit, dateandtime from zoo\_show where nname = 'add any constraints here' ORDER BY nname DESC;

# exhibit

SELECT AH, SIZE, NumAnimals, WF

from (SELECT animalexhibit as AH, S.size AS SIZE, count(\*) AS NumAnimals, S.waterfeature AS WF

from (select animalexhibit, size, waterfeature

from (exhibit AS A JOIN animal B)

where A.nname = B.animalexhibit)

AS S

GROUP BY animalexhibit)

AS L where AH = 'and any constraints' ORDER BY AH ASC ;

SELECT L.A, L.B, L.C

FROM (SELECT nname A, dateandtime B, count(nname) C

from exhibitvisits

WHERE email = ''

GROUP BY (nname, dateandtime))

AS L

WHERE exhibitvisits.nname = '';

SELECT A.nname, A.dateandtime, B.count FROM exhibitvisits A INNER JOIN ((SELECT nname, count(\*) as count FROM exhibitvisits GROUP BY nname)AS B)on A.nname = B.nname WHERE email = 'visitor1@visitor';

## admin======================================================================================

#view visitor

SELECT username, email FROM visitor;

SELECT username, email from visitor ORDER BY email ASC;

DELETE FROM zoo\_user where email = '';

#view staff

SELECT username, email FROM staff;

SELECT username, email from staff ORDER BY email ASC;

DELETE FROM zoo\_user where email = '';

#view animal

SELECT nname, species, animalexhibit, age, animal\_type from animal where nname = 'add any constraints';

DELETE from animal where nname = '' and species = '';

#view show

DELETE FROM zoo\_show where nname = '' and dateandtime = '';

SELECT nname, showexhibit, dateandtime from zoo\_show where nname = 'add any constraints';

#add show

INSERT INTO zoo\_show(nname, dateandtime, showexhibit, showstaff, showdate) values ('','','','','');

#add animal

INSERT into animal(nname, species, animal\_type, age, animalexhibit) values('','','',0,'');

## visitor======================================================================================

#search exhibit

SELECT nname, species from animal where animalexhibit = '';

SELECT nname, species from animal where nname = '' ORDER BY nname ASC;

INSERT into exhibitvisits(email, nname, dateandtime) values ('','','');

SELECT AH, SIZE, NumAnimals, WF

from (SELECT animalexhibit as AH, S.size AS SIZE, count(\*) AS NumAnimals, S.waterfeature AS WF

from (select animalexhibit, size, waterfeature

from (exhibit AS A JOIN animal B)

where A.nname = B.animalexhibit)

AS S

GROUP BY animalexhibit)

AS L

where AH = '';

#search shows

SELECT nname, showexhibit, dateandtime from zoo\_show where nname = '';

INSERT INTO showvisits(email, nname, dateandtime) values ('','','');

#search animals

SELECT nname, species, animalexhibit, age, animal\_type from animal where nname = '';

#view exhibit history

SELECT L.A, L.B, L.C

FROM (SELECT nname A, dateandtime B, count(\*) C

from exhibitvisits

GROUP BY nname, dateandtime)

AS L

where L.A = '';

#show history

SELECT nname, showexhibit, dateandtime from zoo\_show where nname = '';

## visitor======================================================================================

#search animal

SELECT email, notetext, notetime FROM note WHERE nname = '';

SELECT nname, species, animalexhibit, age, animal\_type FROM animal where nname = '';

SELECT email, notetext, notetime FROM note where email = '' ORDER BY email ASC;

#view animal

SELECT nname, showexhibit, dateandtime FROM zoo\_show where showstaff = '';

SELECT B.nname, A.dateandtime, B.nname

FROM (showvisits A JOIN zoo\_show B)

WHERE A.nname = B.nname and A.email = '';

SELECT \*

FROM exhibitvisits

WHERE email = '';

### populate=========================================

INSERT INTO zoo\_user(email, username, ppassword, usertype) VALUES ('rqin37@gatech.edu', 'rqin37', MD5('12345678'),'admin');

INSERT INTO admin(email, username) VALUES ('rqin37@gatech.edu', 'rqin37');

INSERT INTO exhibit(nname, waterfeature, size) VALUES ('Pacific', TRUE , 850);

INSERT INTO exhibit(nname, waterfeature, size) VALUES ('Jungle', FALSE , 600);

INSERT INTO exhibit(nname, waterfeature, size) VALUES ('Sahara', FALSE , 1000);

INSERT INTO exhibit(nname, waterfeature, size) VALUES ('Mountainous', FALSE , 1200);

INSERT INTO exhibit(nname, waterfeature, size) VALUES ('Birds', TRUE , 1000);

import tkinter as tk

from tkinter import \*

from tkinter import messagebox

import tkinter.ttk as ttk

import pymysql

import hashlib

import datetime

import itertools

import re

# get pymysql connection

db = pymysql.connect(host='academic-mysql.cc.gatech.edu',

passwd='gCi5ozqX', user='cs4400\_group22', db='cs4400\_group22')

cursor = db.cursor()

class GUI:

def \_\_init\_\_(self):

self.toLogin()

# after login, user GUI.user variable is created. if running just a one component, create a user handle

## note:

##1. constraints: check(<statement>)

##2. sql build table recheck

#========================================================================

# Login

#========================================================================

##LOGING IN AND CREATING NEW USER--------------------------------------

def toLogin(self):

"""Main login screen"""

# Main window

self.main = Tk()

self.main.title("User Login")

# label/entry frame

win = Frame(self.main)

win.pack()

# labels

Label(win, text="Username:").grid(row=1, column=0, sticky=E, pady=5)

Label(win, text="Password:").grid(row=2, column=0, sticky=E, pady=5)

# Entries

self.username\_entry = Entry(win, width=50)

self.username\_entry.grid(row=1, column=1, padx=5)

self.password\_entry = Entry(win, show="\*", width=50)

self.password\_entry.grid(row=2, column=1, padx=5)

# button frames

buttonframe = Frame(self.main)

buttonframe.pack(anchor=E)

# buttons

Button(buttonframe, text="Login", command=self.connect).pack(side=RIGHT)

Button(buttonframe, text="Register",

command=self.toRegister).pack(side=RIGHT)

# mainloop

self.main.mainloop()

def toRegister(self):

"""Register User Screen"""

# creates new window

self.main.destroy()

self.reg = Tk()

self.reg.title("Create New Account")

# label/entries frame

reg\_frame = Frame(self.reg)

reg\_frame.pack()

# labels

Label(reg\_frame, text="Username:").grid(

row=2, column=0, sticky=E, pady=5)

Label(reg\_frame, text="Password:").grid(

row=3, column=0, sticky=E, pady=5)

Label(reg\_frame, text="Confirm Password:").grid(

row=4, column=0, sticky=E, pady=5)

Label(reg\_frame, text="Email:").grid(

row=5, column=0, sticky=E, pady=5)

Label(reg\_frame, text="Type:").grid(

row=6, column=0, sticky=E)

# username,password,password\_confirm,email entries

self.username = Entry(reg\_frame, width=50)

self.username.grid(row=2, column=1, padx=5)

self.password = Entry(reg\_frame, show="\*", width=50)

self.password.grid(row=3, column=1, padx=5)

self.password\_confirm = Entry(reg\_frame, show="\*", width=50)

self.password\_confirm.grid(row=4, column=1, padx=5)

self.email\_entry = Entry(reg\_frame, width=50)

self.email\_entry.grid(row=5, column=1, padx=5)

self.userVar = StringVar()

self.Types = {"staff", "visitor"}

self.userVar.set("staff")

self.userType = OptionMenu(reg\_frame, self.userVar, \*self.Types)

self.userType.grid(row = 6, column = 1, sticky = "w")

# button frame

buttonframe = Frame(self.reg)

buttonframe.pack(anchor=E)

# buttons

Button(buttonframe, text="Register",

command=self.registerconfirm).pack(side=RIGHT)

Button(buttonframe, text="Cancel",

command=self.reg\_cancel).pack(side=RIGHT)

# mainloop

self.reg.mainloop()

def registerconfirm(self):

"""Helper function for toRegister, does actual action of registering and checking requirements"""

error = False

# check all fields filled in

if self.username.get() is '' or self.password.get() is '' or self.password\_confirm.get() is '' or self.email\_entry.get() is '':

messagebox.showerror(title="Warning", message="All fields are required")

error = True

# check that passwords are equal

if self.password.get() != self.password\_confirm.get():

messagebox.showerror(title="Password Error", message="Passwords must match")

error = True

# check password is valid

if len(self.password.get()) < 8:

messagebox.showerror(title="Password Error", message="Password must be more than 8 characters")

error = True

emailMatching = re.search('^([A-Z, a-z, 0-9]+)@([A-Z, a-z, 0-9]+)(\.)([A-z, a-z, 0-9]+)', self.email\_entry.get())

if emailMatching is None:

error = True

messagebox.showerror(title="Error", message="Invalid Email address!")

if error is False:

# check that username and email doesn't exist

sql = "SELECT email, username FROM zoo\_user where email = \'%s\' and username = \'%s\'" % (self.email\_entry.get(), self.username.get())

cursor.execute(sql)

result = cursor.fetchall()

if (len(result) != 0):

messagebox.showerror(title="Error", message="This username or email already exists")

try:

print (self.password.get())

sql = "INSERT INTO zoo\_user(email, username, ppassword, usertype) VALUES (\'%s\', \'%s\', MD5(\'%s\'), \'%s\')" % (self.email\_entry.get(), self.username.get(), self.password.get(), self.userVar.get())

cursor.execute(sql)

if (self.userVar.get() == "staff"):

sql = "INSERT INTO staff(email, username) VALUES ('" + self.email\_entry.get() + "','" + self.username.get() + "');"

else:

sql = "INSERT INTO visitor(email, username) VALUES ('" + self.email\_entry.get() + "','" + self.username.get() + "');"

cursor.execute(sql)

db.commit()

self.reg.destroy()

self.toLogin()

except:

messagebox.showerror(title='Unknown Error', message='Invalid inputs')

def reg\_cancel(self):

"""Helper function for toRegister, cancels registration and returns to main login scree"""

self.reg.destroy()

self.toLogin()

def connect(self):

"""Helper funtion for toLogin, logs in to system"""

self.user = self.username\_entry.get()

string\_user = self.username\_entry.get()

string\_pw = self.password\_entry.get()

sql = "SELECT username, ppassword, usertype FROM zoo\_user WHERE username=\'%s\' AND ppassword=MD5(\'%s\')" % (string\_user, string\_pw)

cursor.execute(sql)

isUser = cursor.fetchone()

if isUser:

sql\_email = "SELECT email from zoo\_user where username = \'%s\'" % (self.user)

cursor.execute(sql\_email)

self.email = cursor.fetchone()[0]

print(self.email)

if isUser[2] == 'admin':

self.main.destroy()

self.adminMenu()

elif isUser[2] == 'staff':

self.main.destroy()

self.staffMenu()

else:

self.main.destroy()

self.visitorMenu()

else:

messagebox.showerror(

title="Error", message="Invalid username/password combination.")

##Go different screen--------------------------------------------------

def adminMenu(self):

self.main = Tk()

self.main.title('ADMIN')

Label(self.main, text = 'Welcome, Admin ' +

self.user).grid(row = 0, column = 1, columnspan = 5)

Button(self.main, text='View Visitors', command = self.admin\_view\_visitors, width=20).grid(row=1, column=0, columnspan=3, sticky=W, pady=15, padx = 30)

Button(self.main, text='View Staffs', command = self.admin\_view\_staffs, width=20).grid(row=2, column=0, columnspan=3, sticky=W, pady=15, padx = 30)

Button(self.main, text='View Animals', command = self.admin\_view\_animals, width=20).grid(row=3, column=0, columnspan=3, sticky=W, pady=15, padx = 30)

Button(self.main, text='View Shows', command = self.admin\_view\_shows, width=20).grid(row=1, column=5, columnspan=5, sticky=W, pady=15, padx = 30)

Button(self.main, text='Add Shows', command = self.admin\_add\_shows, width=20).grid(row=2, column=5, columnspan=5, sticky=W, pady=15, padx = 30)

Button(self.main, text='Log out', command = self.admin\_log\_out, width=20).grid(row=4, column=5, columnspan=5, sticky=W, pady=15, padx = 30)

Button(self.main, text='Add Animal', command = self.admin\_add\_animal, width=20).grid(row=3, column=5, columnspan=5, sticky=W, pady=15, padx = 30)

self.main.mainloop()

def staffMenu(self):

self.main = Tk()

self.main.title('STAFF')

Label(self.main, text='Welcome, Staff ' + self.user).grid(row=0, column=3, columnspan=5, sticky=W, pady=15, padx = 30)

Button(self.main, text='Search Animals', command = self.staff\_search\_animals).grid(row=1, column=0, columnspan=5, sticky=W, pady=15, padx = 30)

Button(self.main, text='View Shows', command = self.staff\_view\_shows).grid(row=2, column=0, columnspan=5, sticky=W, pady=15, padx = 30)

Button(self.main, text='Log out', command = self.staff\_log\_out).grid(row=3, column=0, columnspan=5, sticky=W, pady=15, padx = 30)

self.main.mainloop()

def visitorMenu(self):

self.main = Tk()

self.main.title('VISITOR')

Label(self.main, text='Welcome, Visitor ' +

self.user).grid(row=0, column=1, columnspan=5)

Button(self.main, text='Search Exhibits', command = self.visitor\_search\_exhibits, width=20).grid(row=1, column=0, columnspan=3, sticky=W, pady=15, padx = 30)

Button(self.main, text='Search Shows', command = self.visitor\_search\_shows, width=20).grid(row=2, column=0, columnspan=3, sticky=W, pady=15, padx = 30)

Button(self.main, text='Search Animals', command = self.visitor\_search\_animals, width=20).grid(row=3, column=0, columnspan=3, sticky=W, pady=15, padx = 30)

Button(self.main, text='View exhibit history', command = self.visitor\_view\_exhibit\_history, width=20).grid(row=1, column=5, columnspan=5, sticky=W, pady=15, padx = 30)

Button(self.main, text='View show history', command = self.visitor\_view\_show\_history, width=20).grid(row=2, column=5, columnspan=5, sticky=W, pady=15, padx = 30)

Button(self.main, text='Log Out', command = self.visitor\_log\_out, width=20).grid(row=3, column=5, columnspan=5, sticky=W, pady=15, padx = 30)

self.main.mainloop()

#========================================================================

# sort

#========================================================================

## Animal-----------------------------------------

def sort\_animal\_name\_asc(self):

print ('animal name asc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

Label(temp\_view, text='type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W, padx=10)

sql = "SELECT nname, species, animalexhibit, age, animal\_type from animal %s ORDER BY nname ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=row[4], padx=10).grid(row=i, column=4, sticky=W)

i += 1

def sort\_animal\_name\_desc(self):

print ('animal name desc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

Label(temp\_view, text='type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W, padx=10)

sql = "SELECT nname, species, animalexhibit, age, animal\_type from animal %s ORDER BY nname DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=row[4], padx=10).grid(row=i, column=4, sticky=W)

i += 1

def sort\_animal\_species\_asc(self):

print ('animal species asc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

Label(temp\_view, text='type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W, padx=10)

sql = "SELECT nname, species, animalexhibit, age, animal\_type from animal %s ORDER BY species ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=row[4], padx=10).grid(row=i, column=4, sticky=W)

i += 1

def sort\_animal\_species\_desc(self):

print ('animal species desc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

Label(temp\_view, text='type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W, padx=10)

sql = "SELECT nname, species, animalexhibit, age, animal\_type from animal %s ORDER BY species DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=row[4], padx=10).grid(row=i, column=4, sticky=W)

i += 1

def sort\_animal\_exhibit\_asc(self):

print ('animal exhibit asc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

Label(temp\_view, text='type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W, padx=10)

sql = "SELECT nname, species, animalexhibit, age, animal\_type from animal %s ORDER BY animalexhibit ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=row[4], padx=10).grid(row=i, column=4, sticky=W)

i += 1

def sort\_animal\_exhibit\_desc(self):

print ('animal exhibit desc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

Label(temp\_view, text='type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W, padx=10)

sql = "SELECT nname, species, animalexhibit, age, animal\_type from animal %s ORDER BY animalexhibit DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=row[4], padx=10).grid(row=i, column=4, sticky=W)

i += 1

def sort\_animal\_age\_asc(self):

print ('animal age asc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

Label(temp\_view, text='type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W, padx=10)

sql = "SELECT nname, species, animalexhibit, age, animal\_type from animal %s ORDER BY age ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=row[4], padx=10).grid(row=i, column=4, sticky=W)

i += 1

def sort\_animal\_age\_desc(self):

print ('animal age desc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

Label(temp\_view, text='type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W, padx=10)

sql = "SELECT nname, species, animalexhibit, age, animal\_type from animal %s ORDER BY age DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=row[4], padx=10).grid(row=i, column=4, sticky=W)

i += 1

def sort\_animal\_type\_asc(self):

print ('animal type asc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

Label(temp\_view, text='type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W, padx=10)

sql = "SELECT nname, species, animalexhibit, age, animal\_type from animal %s ORDER BY animal\_type ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=row[4], padx=10).grid(row=i, column=4, sticky=W)

i += 1

def sort\_animal\_type\_desc(self):

print ('animal type desc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

Label(temp\_view, text='type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W, padx=10)

sql = "SELECT nname, species, animalexhibit, age, animal\_type from animal %s ORDER BY animal\_type DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=row[4], padx=10).grid(row=i, column=4, sticky=W)

i += 1

## Show-----------------------------------------

def sort\_show\_name\_asc(self):

temp\_view = Tk()

temp\_view.title('Show')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Date', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

sql = "SELECT nname, showexhibit, dateandtime from zoo\_show %s ORDER BY nname ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

i += 1

def sort\_show\_name\_desc(self):

temp\_view = Tk()

temp\_view.title('Show')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Date', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

sql = "SELECT nname, showexhibit, dateandtime from zoo\_show %s ORDER BY nname DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

i += 1

def sort\_show\_exhibit\_asc(self):

temp\_view = Tk()

temp\_view.title('Show')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Date', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

sql = "SELECT nname, showexhibit, dateandtime from zoo\_show %s ORDER BY showexhibit ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

i += 1

def sort\_show\_exhibit\_desc(self):

temp\_view = Tk()

temp\_view.title('Show')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Date', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

sql = "SELECT nname, showexhibit, dateandtime from zoo\_show %s ORDER BY showexhibit DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

i += 1

def sort\_show\_time\_asc(self):

temp\_view = Tk()

temp\_view.title('Show')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Date', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

sql = "SELECT nname, showexhibit, dateandtime from zoo\_show %s ORDER BY dateandtime ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

i += 1

def sort\_show\_time\_desc(self):

temp\_view = Tk()

temp\_view.title('Show')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Date', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

sql = "SELECT nname, showexhibit, dateandtime from zoo\_show %s ORDER BY dateandtime DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

i += 1

## Exhibit-----------------------------------------

def sort\_exhibit\_name\_asc(self):

temp\_view = Tk()

temp\_view.title('Exhibit')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='size', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Number Animals', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='Water', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

sql = "SELECT AH, SIZE, NumAnimals, WF from (SELECT animalexhibit as AH, S.size AS SIZE, count(\*) AS NumAnimals, S.waterfeature AS WF from (select animalexhibit, size, waterfeature from (exhibit AS A JOIN animal B) where A.nname = B.animalexhibit) AS S GROUP BY animalexhibit) AS L %s ORDER BY AH ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

i += 1

def sort\_exhibit\_name\_desc(self):

temp\_view = Tk()

temp\_view.title('Exhibit')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='size', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Number Animals', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='Water', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

sql = "SELECT AH, SIZE, NumAnimals, WF from (SELECT animalexhibit as AH, S.size AS SIZE, count(\*) AS NumAnimals, S.waterfeature AS WF from (select animalexhibit, size, waterfeature from (exhibit AS A JOIN animal B) where A.nname = B.animalexhibit) AS S GROUP BY animalexhibit) AS L %s ORDER BY AH DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

i += 1

def sort\_exhibit\_size\_asc(self):

temp\_view = Tk()

temp\_view.title('Exhibit')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='size', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Number Animals', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='Water', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

sql = "SELECT AH, SIZE, NumAnimals, WF from (SELECT animalexhibit as AH, S.size AS SIZE, count(\*) AS NumAnimals, S.waterfeature AS WF from (select animalexhibit, size, waterfeature from (exhibit AS A JOIN animal B) where A.nname = B.animalexhibit) AS S GROUP BY animalexhibit) AS L %s ORDER BY SIZE ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

i += 1

def sort\_exhibit\_size\_desc(self):

temp\_view = Tk()

temp\_view.title('Exhibit')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='size', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Number Animals', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='Water', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

sql = "SELECT AH, SIZE, NumAnimals, WF from (SELECT animalexhibit as AH, S.size AS SIZE, count(\*) AS NumAnimals, S.waterfeature AS WF from (select animalexhibit, size, waterfeature from (exhibit AS A JOIN animal B) where A.nname = B.animalexhibit) AS S GROUP BY animalexhibit) AS L %s ORDER BY SIZE DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

i += 1

def sort\_exhibit\_numAnimals\_asc(self):

temp\_view = Tk()

temp\_view.title('Exhibit')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='size', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Number Animals', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='Water', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

sql = "SELECT AH, SIZE, NumAnimals, WF from (SELECT animalexhibit as AH, S.size AS SIZE, count(\*) AS NumAnimals, S.waterfeature AS WF from (select animalexhibit, size, waterfeature from (exhibit AS A JOIN animal B) where A.nname = B.animalexhibit) AS S GROUP BY animalexhibit) AS L %s ORDER BY NumAnimals ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

i += 1

def sort\_exhibit\_numAnimals\_desc(self):

temp\_view = Tk()

temp\_view.title('Exhibit')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='size', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Number Animals', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='Water', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

sql = "SELECT AH, SIZE, NumAnimals, WF from (SELECT animalexhibit as AH, S.size AS SIZE, count(\*) AS NumAnimals, S.waterfeature AS WF from (select animalexhibit, size, waterfeature from (exhibit AS A JOIN animal B) where A.nname = B.animalexhibit) AS S GROUP BY animalexhibit) AS L %s ORDER BY NumAnimals DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

i += 1

def sort\_exhibit\_waterFeature\_asc(self):

temp\_view = Tk()

temp\_view.title('Exhibit')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='size', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Number Animals', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='Water', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

sql = "SELECT AH, SIZE, NumAnimals, WF from (SELECT animalexhibit as AH, S.size AS SIZE, count(\*) AS NumAnimals, S.waterfeature AS WF from (select animalexhibit, size, waterfeature from (exhibit AS A JOIN animal B) where A.nname = B.animalexhibit) AS S GROUP BY animalexhibit) AS L %s ORDER BY WF ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

i += 1

def sort\_exhibit\_waterFeature\_desc(self):

temp\_view = Tk()

temp\_view.title('Exhibit')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='size', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Number Animals', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='Water', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

sql = "SELECT AH, SIZE, NumAnimals, WF from (SELECT animalexhibit as AH, S.size AS SIZE, count(\*) AS NumAnimals, S.waterfeature AS WF from (select animalexhibit, size, waterfeature from (exhibit AS A JOIN animal B) where A.nname = B.animalexhibit) AS S GROUP BY animalexhibit) AS L %s ORDER BY WF DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

i += 1

#========================================================================

# admin

#========================================================================

##=======================(admin) view visitors=========================##

def admin\_view\_visitors(self):

self.view\_admin\_view\_visitors = Tk()

self.view\_admin\_view\_visitors.title('View Visitors')

view\_admin\_frame = Frame(self.view\_admin\_view\_visitors)

view\_admin\_frame.pack()

#delete visitor

self.delete\_visitor = Entry(view\_admin\_frame, width=30, textvariable=StringVar(view\_admin\_frame, value = 'email'))

self.delete\_visitor.grid(row=1, column = 0, padx = 5)

button = Button(view\_admin\_frame, text = "delete", width=10, command = self.helper\_admin\_view\_visitors)

button.grid(row=1,column=1)

# backend code

sql = "SELECT username, email FROM visitor"

cursor.execute(sql)

# reivew = cursor.fetchall()

review = []

for row in cursor:

review.append(row)

# insert chart headers

Label(view\_admin\_frame, text='username', font="Lucida 12 bold ").grid(

row=2, column=0, sticky=W)

Label(view\_admin\_frame, text='email', font="Lucida 12 bold ").grid(

row=2, column=1, sticky=W)

i = 3

for row in review:

Label(view\_admin\_frame, text=row[0]).grid(row=i, column=0, sticky=W)

Label(view\_admin\_frame, text=row[1]).grid(row=i, column=1, sticky=W)

i += 1

# sort

button = Button(view\_admin\_frame, text = "Email ASC", width=15, command = self.helper\_visitor\_email\_asc)

button.grid(row=i,column=1, sticky=W)

button = Button(view\_admin\_frame, text = "Email DESC", width=15, command = self.helper\_visitor\_email\_desc)

button.grid(row=i + 1,column=1, sticky=W)

button = Button(view\_admin\_frame, text = "Username ASC", width=15, command = self.helper\_visitor\_username\_asc)

button.grid(row=i,column=0, sticky=W)

button = Button(view\_admin\_frame, text = "Username DESC", width=15, command = self.helper\_visitor\_username\_desc)

button.grid(row=i + 1,column=0, sticky=W)

#helpers

def helper\_visitor\_email\_asc(self):

view = Tk()

view.title('Sorting Ascending by Email')

sql = "SELECT username, email from visitor ORDER BY email ASC"

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

# insert chart headers

Label(view, text='username', font="Lucida 12 bold ").grid(

row=2, column=0, sticky=W)

Label(view, text='email', font="Lucida 12 bold ").grid(

row=2, column=1, sticky=W)

i = 3

for row in review:

Label(view, text=row[0]).grid(row=i, column=0, sticky=W)

Label(view, text=row[1]).grid(row=i, column=1, sticky=W)

i += 1

def helper\_visitor\_email\_desc(self):

view = Tk()

view.title('Sorting Descending by Email')

sql = "SELECT username, email from visitor ORDER BY email DESC"

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

# insert chart headers

Label(view, text='username', font="Lucida 12 bold ").grid(

row=2, column=0, sticky=W)

Label(view, text='email', font="Lucida 12 bold ").grid(

row=2, column=1, sticky=W)

i = 3

for row in review:

Label(view, text=row[0]).grid(row=i, column=0, sticky=W)

Label(view, text=row[1]).grid(row=i, column=1, sticky=W)

i += 1

def helper\_visitor\_username\_asc(self):

view = Tk()

view.title('Sorting Ascending by Username')

sql = "SELECT username, email from visitor ORDER BY username ASC"

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

# insert chart headers

Label(view, text='username', font="Lucida 12 bold ").grid(

row=2, column=0, sticky=W)

Label(view, text='email', font="Lucida 12 bold ").grid(

row=2, column=1, sticky=W)

i = 3

for row in review:

Label(view, text=row[0]).grid(row=i, column=0, sticky=W)

Label(view, text=row[1]).grid(row=i, column=1, sticky=W)

i += 1

def helper\_visitor\_username\_desc(self):

view = Tk()

view.title('Sorting Decending by Username')

sql = "SELECT username, email from visitor ORDER BY username DESC"

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

# insert chart headers

Label(view, text='username', font="Lucida 12 bold ").grid(

row=2, column=0, sticky=W)

Label(view, text='email', font="Lucida 12 bold ").grid(

row=2, column=1, sticky=W)

i = 3

for row in review:

Label(view, text=row[0]).grid(row=i, column=0, sticky=W)

Label(view, text=row[1]).grid(row=i, column=1, sticky=W)

i += 1

def helper\_admin\_view\_visitors(self):

sql = "DELETE FROM zoo\_user where email = \'%s\'" % (self.delete\_visitor.get())

cursor.execute(sql)

self.view\_admin\_view\_visitors.destroy()

db.commit()

##=======================(admin) view staffs===========================##

def admin\_view\_staffs(self):

self.view\_admin\_view\_staffs = Tk()

self.view\_admin\_view\_staffs.title('View Staffs')

view = Frame(self.view\_admin\_view\_staffs)

view.pack()

#delete staff

self.delete\_staff = Entry(view, width=30, textvariable=StringVar(view, value='email'))

self.delete\_staff.grid(row=1,column=0,padx=5)

button = Button(view, text="delete", width=10, command=self.helper\_admin\_view\_staffs)

button.grid(row=1,column=1)

sql = "SELECT username, email FROM staff"

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

# insert chart headers

Label(view, text='username', font="Lucida 12 bold ").grid(

row=2, column=0, sticky=W)

Label(view, text='email', font="Lucida 12 bold ").grid(

row=2, column=1, sticky=W)

i = 3

for row in review:

Label(view, text=row[0]).grid(row=i, column=0, sticky=W)

Label(view, text=row[1]).grid(row=i, column=1, sticky=W)

i += 1

button = Button(view, text = "Email ASC", width=15, command = self.helper\_staff\_email\_asc)

button.grid(row=i,column=1, sticky=W)

button = Button(view, text = "Email DESC", width=15, command = self.helper\_staff\_email\_desc)

button.grid(row=i + 1,column=1, sticky=W)

button = Button(view, text = "Username ASC", width=15, command = self.helper\_staff\_username\_asc)

button.grid(row=i,column=0, sticky=W)

button = Button(view, text = "Username DESC", width=15, command = self.helper\_staff\_username\_desc)

button.grid(row=i + 1,column=0, sticky=W)

#helpers

def helper\_staff\_email\_asc(self):

view = Tk()

view.title('Sorting Ascending by Email')

sql = "SELECT username, email from staff ORDER BY email ASC"

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

# insert chart headers

Label(view, text='username', font="Lucida 12 bold ").grid(

row=2, column=0, sticky=W)

Label(view, text='email', font="Lucida 12 bold ").grid(

row=2, column=1, sticky=W)

i = 3

for row in review:

Label(view, text=row[0]).grid(row=i, column=0, sticky=W)

Label(view, text=row[1]).grid(row=i, column=1, sticky=W)

i += 1

def helper\_staff\_email\_desc(self):

view = Tk()

view.title('Sorting Descending by Email')

sql = "SELECT username, email from staff ORDER BY email DESC"

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

# insert chart headers

Label(view, text='username', font="Lucida 12 bold ").grid(

row=2, column=0, sticky=W)

Label(view, text='email', font="Lucida 12 bold ").grid(

row=2, column=1, sticky=W)

i = 3

for row in review:

Label(view, text=row[0]).grid(row=i, column=0, sticky=W)

Label(view, text=row[1]).grid(row=i, column=1, sticky=W)

i += 1

def helper\_staff\_username\_asc(self):

view = Tk()

view.title('Sorting Ascending by Username')

sql = "SELECT username, email from staff ORDER BY username ASC"

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

# insert chart headers

Label(view, text='username', font="Lucida 12 bold ").grid(

row=2, column=0, sticky=W)

Label(view, text='email', font="Lucida 12 bold ").grid(

row=2, column=1, sticky=W)

i = 3

for row in review:

Label(view, text=row[0]).grid(row=i, column=0, sticky=W)

Label(view, text=row[1]).grid(row=i, column=1, sticky=W)

i += 1

def helper\_staff\_username\_desc(self):

view = Tk()

view.title('Sorting Decending by Username')

sql = "SELECT username, email from staff ORDER BY username DESC"

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

# insert chart headers

Label(view, text='username', font="Lucida 12 bold ").grid(

row=2, column=0, sticky=W)

Label(view, text='email', font="Lucida 12 bold ").grid(

row=2, column=1, sticky=W)

i = 3

for row in review:

Label(view, text=row[0]).grid(row=i, column=0, sticky=W)

Label(view, text=row[1]).grid(row=i, column=1, sticky=W)

i += 1

def helper\_admin\_view\_staffs(self):

sql = "DELETE FROM zoo\_user where email = \'%s\'" % (self.delete\_staff.get())

cursor.execute(sql)

self.view\_admin\_view\_staffs.destroy()

db.commit()

##=======================(admin) view animals==========================##

def admin\_view\_animals(self):

self.view\_admin\_view\_animals = Tk()

self.view\_admin\_view\_animals.title("View Animals")

view = Frame(self.view\_admin\_view\_animals)

view.pack()

#animal search

Label(view, text='name', font="Lucida 12 bold ", width=15).grid(

row=1, column=0, sticky=W)

self.admin\_view\_animals\_name = Entry(view, width=20, textvariable=StringVar(view, value='NAME'))

self.admin\_view\_animals\_name.grid(row=2,column=0)

Label(view, text='species', font="Lucida 12 bold ", width=15).grid(

row=1, column=1, sticky=W)

self.admin\_view\_animals\_species = Entry(view, width=20, textvariable=StringVar(view, value='SPECIES'))

self.admin\_view\_animals\_species.grid(row=2,column=1)

Label(view, text='exhibit', font="Lucida 12 bold ", width=15).grid(

row=1, column=2, sticky=W)

self.exhibit\_Types2 = {'Jungle','Pacific','Sahara','Mountainous', 'Birds', ''}

self.admin\_view\_animals\_exhibit\_entry = StringVar(view, value='')

self.admin\_view\_animals\_exhibit = OptionMenu(view, self.admin\_view\_animals\_exhibit\_entry, \*self.exhibit\_Types2)

self.admin\_view\_animals\_exhibit.grid(row=2,column=2)

Label(view, text='age', font="Lucida 12 bold ", width=15).grid(

row=1, column=3, sticky=W)

self.admin\_view\_animal\_age\_min = Entry(view, width=20, textvariable=IntVar(view, value=0))

self.admin\_view\_animal\_age\_min.grid(row=2,column=3,pady=5)

self.admin\_view\_animal\_age\_max = Entry(view, width=20, textvariable=IntVar(view, value=99999999))

self.admin\_view\_animal\_age\_max.grid(row=3,column=3,pady=5)

Label(view, text='type', font="Lucida 12 bold ", width=15).grid(

row=1, column=4, sticky=W)

self.animal\_types = ['Mammal', 'Bird', 'Amphibian', 'Reptile', 'Fish', 'Invertebrate', '']

self.animal\_types\_entry = StringVar(view, value='')

self.admin\_view\_animals\_type = OptionMenu(view, self.animal\_types\_entry, \*self.animal\_types)

self.admin\_view\_animals\_type.grid(row=2,column=4)

button = Button(view, text='search', width = 20, command=self.helper\_admin\_view\_animals)

button.grid(row=3,column=0)

# animal delete

self.admin\_view\_animals\_name\_delete = Entry(view, width=20, textvariable=StringVar(view, value='NAME'))

self.admin\_view\_animals\_name\_delete.grid(row=6,column=0)

self.admin\_view\_animals\_species\_delete = Entry(view, width=20, textvariable=StringVar(view, value='SPECIES'))

self.admin\_view\_animals\_species\_delete.grid(row=6,column=1)

button = Button(view, text='delete', width = 20, command=self.helper\_delete\_admin\_view\_animals)

button.grid(row=6,column=2)

db.commit()

def helper\_admin\_view\_animals(self):

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

Label(temp\_view, text='type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W, padx=10)

name = self.admin\_view\_animals\_name.get()

species = self.admin\_view\_animals\_species.get()

exhibit = self.admin\_view\_animals\_exhibit\_entry.get()

min\_age = self.admin\_view\_animal\_age\_min.get()

max\_age = self.admin\_view\_animal\_age\_max.get()

animal\_type = self.animal\_types\_entry.get()

clause = ''

whereClauseStart = False

if name != 'NAME':

if not whereClauseStart:

whereClauseStart = True

clause = "where nname = \'%s\'" % (name)

else:

clause += "and nname = \'%s\'" % (name)

if species != 'SPECIES':

if not whereClauseStart:

whereClauseStart = True

clause = "where species = \'%s\'" % (species)

else:

clause += "and species = \'%s\'" % (species)

if animal\_type != '':

if not whereClauseStart:

whereClauseStart = True

clause = "where animal\_type = \'%s\'" % (animal\_type)

else:

clause += "and animal\_type = \'%s\'" % (animal\_type)

if exhibit != '':

if not whereClauseStart:

whereClauseStart = True

clause = "where animalexhibit = \'%s\'" % (exhibit)

else:

clause += "and animalexhibit = \'%s\'" % (exhibit)

if clause == '':

clause = "where age >= %s and age <= %s" % (min\_age, max\_age)

else:

clause += "and age >= %s and age <= %s" % (min\_age, max\_age)

sql = "SELECT nname, species, animalexhibit, age, animal\_type from animal %s" % (clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

self.sql\_admin\_view\_animals\_list = review

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=row[4], padx=10).grid(row=i, column=4, sticky=W)

i += 1

db.commit()

# sort

self.clause = clause

button = Button(temp\_view, text = "Name ASC", width=15, command = self.sort\_animal\_name\_asc)

button.grid(row=i,column=0, sticky=W)

button = Button(temp\_view, text = "Name DESC", width=15, command = self.sort\_animal\_name\_desc)

button.grid(row=i + 1,column=0, sticky=W)

button = Button(temp\_view, text = "Species ASC", width=15, command = self.sort\_animal\_species\_asc)

button.grid(row=i,column=1, sticky=W)

button = Button(temp\_view, text = "Species DESC", width=15, command = self.sort\_animal\_species\_desc)

button.grid(row=i + 1,column=1, sticky=W)

button = Button(temp\_view, text = "Exhibit ASC", width=15, command = self.sort\_animal\_exhibit\_asc)

button.grid(row=i,column=2, sticky=W)

button = Button(temp\_view, text = "Exhibit DESC", width=15, command = self.sort\_animal\_exhibit\_desc)

button.grid(row=i + 1,column=2, sticky=W)

button = Button(temp\_view, text = "Age ASC", width=15, command = self.sort\_animal\_age\_asc)

button.grid(row=i,column=3, sticky=W)

button = Button(temp\_view, text = "Age DESC", width=15, command = self.sort\_animal\_age\_desc)

button.grid(row=i + 1,column=3, sticky=W)

button = Button(temp\_view, text = "Type ASC", width=15, command = self.sort\_animal\_type\_asc)

button.grid(row=i,column=4, sticky=W)

button = Button(temp\_view, text = "Type DESC", width=15, command = self.sort\_animal\_type\_desc)

button.grid(row=i + 1,column=4, sticky=W)

def helper\_delete\_admin\_view\_animals(self):

name\_delete = self.admin\_view\_animals\_name\_delete.get()

species\_delete = self.admin\_view\_animals\_species\_delete.get()

sql = "DELETE from animal where nname = \'%s\' and species = \'%s\'" % (name\_delete, species\_delete)

cursor.execute(sql)

db.commit()

self.view\_admin\_view\_animals.destroy()

##=======================(admin) view shows============================##

def admin\_view\_shows(self):

self.view\_shows = Tk()

self.view\_shows.title('View Shows')

view = Frame(self.view\_shows)

view.pack()

#search

Label(view,text='Name').grid(row=1,column=0, sticky = W)

self.admin\_view\_shows\_searchname = Entry(view, width=20, textvariable=StringVar(view, value = 'NAME'))

self.admin\_view\_shows\_searchname.grid(row=2,column=0)

Label(view,text='Exhibit').grid(row=1,column=1, sticky = W)

self.exhibit\_Types1 = {"Jungle","Pacific","Sahara","Mountainous", "Birds", ""}

self.admin\_view\_shows\_exhibit\_entry = StringVar(view, value='')

self.admin\_view\_shows\_exhibit = OptionMenu(view, self.admin\_view\_shows\_exhibit\_entry, \*self.exhibit\_Types1)

self.admin\_view\_shows\_exhibit.grid(row=2,column=1)

Label(view,text='Date').grid(row=1,column=2, sticky = W)

self.admin\_view\_shows\_date = Entry(view, width = 20, textvariable=StringVar(view, value = 'YYYY-MM-DD'))

self.admin\_view\_shows\_date.grid(row=2,column=2)

button = Button(view, text = "search", width=20, command=self.helper\_admin\_view\_shows\_search)

button.grid(row=3,column=0)

# show delete

self.admin\_view\_show\_name\_delete = Entry(view, width=20, textvariable=StringVar(view, value='NAME'))

self.admin\_view\_show\_name\_delete.grid(row=4,column=0)

self.admin\_view\_show\_dateandtime\_delete = Entry(view, width=20, textvariable=StringVar(view, value='YYYY-MM-DD 00:00:00'))

self.admin\_view\_show\_dateandtime\_delete.grid(row=4,column=1)

button = Button(view, text='delete', width = 20, command=self.helper\_delete\_admin\_view\_show)

button.grid(row=4,column=2)

db.commit()

def helper\_delete\_admin\_view\_show(self):

name = self.admin\_view\_show\_name\_delete.get()

time = self.admin\_view\_show\_dateandtime\_delete.get()

sql = "DELETE FROM zoo\_show where nname = \'%s\' and dateandtime = \'%s\'" % (name, time)

print (sql)

cursor.execute(sql)

db.commit()

self.view\_admin\_show\_result.destroy()

def helper\_admin\_view\_shows\_search(self):

self.view\_admin\_show\_result = Tk()

self.view\_admin\_show\_result.title("Shows")

temp\_view = Frame(self.view\_admin\_show\_result)

temp\_view.pack()

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W)

Label(temp\_view, text='date', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W)

name = self.admin\_view\_shows\_searchname.get()

exhibit = self.admin\_view\_shows\_exhibit\_entry.get()

date = self.admin\_view\_shows\_date.get()

clause = ''

whereClauseStart = False

if name != 'NAME':

if not whereClauseStart:

whereClauseStart = True

clause = "where nname = \'%s\'" %(name)

else:

clause += "and nname = \'%s\'" %(name)

if exhibit != '':

if not whereClauseStart:

whereClauseStart = True

clause = "where showexhibit = \'%s\'" %(exhibit)

else:

clause += "and showexhibit = \'%s\'" %(exhibit)

if date != 'YYYY-MM-DD':

if not whereClauseStart:

whereClauseStart = True

clause = "where showdate = \'%s\'" %(date)

else:

clause += "and showdate = \'%s\'" %(date)

sql = "SELECT nname, showexhibit, dateandtime from zoo\_show %s" % (clause)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

self.clause = clause

i = 2

for row in review:

Label(temp\_view, text=row[0]).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1]).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2]).grid(row=i, column=2, sticky=W)

i += 1

button = Button(temp\_view, text = "Name ASC", width=15, command = self.sort\_show\_name\_asc)

button.grid(row=i,column=0, sticky=W)

button = Button(temp\_view, text = "Name DESC", width=15, command = self.sort\_show\_name\_desc)

button.grid(row=i + 1,column=0, sticky=W)

button = Button(temp\_view, text = "Exhibit ASC", width=15, command = self.sort\_show\_exhibit\_asc)

button.grid(row=i,column=1, sticky=W)

button = Button(temp\_view, text = "Exhibit DESC", width=15, command = self.sort\_show\_exhibit\_desc)

button.grid(row=i + 1,column=1, sticky=W)

button = Button(temp\_view, text = "Date ASC", width=15, command = self.sort\_show\_time\_asc)

button.grid(row=i,column=2, sticky=W)

button = Button(temp\_view, text = "Date DESC", width=15, command = self.sort\_show\_time\_desc)

button.grid(row=i + 1,column=2, sticky=W)

db.commit()

##=======================(admin) add shows=============================##

def admin\_add\_shows(self):

self.add\_shows\_view = Tk()

self.add\_shows\_view.title('Add Shows')

view\_add\_show = Frame(self.add\_shows\_view)

view\_add\_show.pack()

Label(view\_add\_show, text="Name:").grid(

row=1, column=0, sticky=E, pady=5)

self.showname = Entry(view\_add\_show, width = 20)

self.showname.grid(row=1, column = 1, padx=5)

Label(view\_add\_show, text="Exhibit:").grid(

row=2, column=0, sticky=E)

self.userVar1 = StringVar(view\_add\_show, value="Jungle")

self.exhibit\_Types = {"Jungle","Pacific","Sahara","Mountainous", "Birds"}

self.showexhibit = OptionMenu(view\_add\_show, self.userVar1, \*self.exhibit\_Types)

self.showexhibit.grid(row=2, column=1, sticky=W)

Label(view\_add\_show, text="Staff:").grid(

row=3, column=0, sticky=E, pady=5)

sql = "SELECT email FROM staff"

cursor.execute(sql)

staff\_list = cursor.fetchall()

choice = ['']

if (len(staff\_list) != 0) :

for element in staff\_list:

choice.append(element[0])

self.staff\_choice = choice

self.userVar2 = StringVar(view\_add\_show, value="staff name")

self.showstaff = OptionMenu(view\_add\_show, self.userVar2, \*self.staff\_choice)

self.showstaff.grid(row=3, column=1, sticky=W)

Label(view\_add\_show, text="Date:").grid(

row=4, column=0, sticky=E, pady=5)

self.showdata = Entry(view\_add\_show, width = 20, textvariable=StringVar(view\_add\_show, value = 'yy/mm/dd'))

self.showdata.grid(row=4, column = 1, padx=5)

Label(view\_add\_show, text="Time:").grid(

row=5, column=0, sticky=E, pady=5)

self.showtime = Entry(view\_add\_show, width = 20, textvariable=StringVar(view\_add\_show, value = '00:00:00'))

self.showtime.grid(row=5, column = 1, padx=5)

buttonframe = Frame(self.add\_shows\_view)

buttonframe.pack(anchor=E)

Button(buttonframe, command=self.helper\_admin\_add\_shows, text='ADD').pack(side=RIGHT)

def helper\_admin\_add\_shows(self):

val1 = self.showname.get()

print(val1)

val2 = self.showdata.get() + ' ' + self.showtime.get()

print(val2)

val3 = self.userVar1.get()

print(val3)

val4 = self.userVar2.get()

print(val4)

sql = "INSERT INTO zoo\_show(nname, dateandtime, showexhibit, showstaff, showdate) values(\'%s\', \'%s\', \'%s\', \'%s\', \'%s\')" % (val1, val2, val3, val4, self.showdata.get())

print(sql)

cursor.execute(sql)

db.commit()

self.add\_shows\_view.destroy()

##=======================(admin) add animal============================##

def admin\_add\_animal(self):

self.view\_add\_animal = Tk()

self.view\_add\_animal.title('Add animal')

view = Frame(self.view\_add\_animal)

view.pack()

Label(view, text='name', font="Lucida 12 bold ", width=15).grid(

row=1, column=0, sticky=W)

self.admin\_add\_animals\_name = Entry(view, width=20, textvariable=StringVar(view, value='NAME'))

self.admin\_add\_animals\_name.grid(row=2,column=0)

Label(view, text='species', font="Lucida 12 bold ", width=15).grid(

row=1, column=1, sticky=W)

self.admin\_add\_animals\_species = Entry(view, width=20, textvariable=StringVar(view, value='SPECIES'))

self.admin\_add\_animals\_species.grid(row=2,column=1)

Label(view, text='exhibit', font="Lucida 12 bold ", width=15).grid(

row=1, column=2, sticky=W)

self.exhibit\_Types2 = {'Jungle','Pacific','Sahara','Mountainous', 'Birds', ''}

self.admin\_add\_animals\_exhibit\_entry = StringVar(view, value='')

self.admin\_add\_animals\_exhibit = OptionMenu(view, self.admin\_add\_animals\_exhibit\_entry, \*self.exhibit\_Types2)

self.admin\_add\_animals\_exhibit.grid(row=2,column=2)

Label(view, text='age', font="Lucida 12 bold ", width=15).grid(

row=1, column=3, sticky=W)

self.admin\_add\_animal\_age = Entry(view, width=20, textvariable=IntVar(view, value=0))

self.admin\_add\_animal\_age.grid(row=2,column=3,pady=5)

Label(view, text='type', font="Lucida 12 bold ", width=15).grid(

row=1, column=4, sticky=W)

self.animal\_types = ['Mammal', 'Bird', 'Amphibian', 'Reptile', 'Fish', 'Invertebrate', '']

self.animal\_types\_entry = StringVar(view, value='')

self.admin\_add\_animals\_type = OptionMenu(view, self.animal\_types\_entry, \*self.animal\_types)

self.admin\_add\_animals\_type.grid(row=2,column=4)

button = Button(view, text='add', width = 20, command=self.helper\_admin\_add\_animals)

button.grid(row=3,column=0)

db.commit()

def helper\_admin\_add\_animals(self):

name = self.admin\_add\_animals\_name.get()

species = self.admin\_add\_animals\_species.get()

exhibit = self.admin\_add\_animals\_exhibit\_entry.get()

print(exhibit)

age = self.admin\_add\_animal\_age.get()

animal\_type =self.animal\_types\_entry.get()

sql = "INSERT into animal(nname, species, animal\_type, age, animalexhibit) values (\'%s\',\'%s\',\'%s\',\'%s\',\'%s\')" % (name, species, animal\_type, age, exhibit)

print (sql)

if int(age) >= 0:

cursor.execute(sql)

db.commit()

self.view\_add\_animal.destroy()

##=======================(admin) log out===============================##

def admin\_log\_out(self):

self.main.destroy()

#========================================================================

# visitor

#========================================================================

##=======================(visitor) search exhibits=====================##

def visitor\_search\_exhibits(self):

self.view\_visitor\_search\_exhibits = Tk()

self.view\_visitor\_search\_exhibits.title('search exhibits')

view = Frame(self.view\_visitor\_search\_exhibits)

view.pack()

#exhibit search

Label(view, text='Name', font="Lucida 12 bold ", width=15).grid(

row=1, column=0, sticky=W)

self.visitor\_search\_exhibit\_name = Entry(view, width=20, textvariable=StringVar(view, value='NAME'))

self.visitor\_search\_exhibit\_name.grid(row=2,column=0)

Label(view, text='Water Feature', font="Lucida 12 bold ", width=15).grid(

row=1, column=1, sticky=W)

self.visitor\_search\_exhibit\_waterFeature\_Entry = StringVar(view, value = '')

self.water\_feature\_option = [True, False, '']

self.visitor\_search\_exhibit\_waterFeature = OptionMenu(view, self.visitor\_search\_exhibit\_waterFeature\_Entry, \*self.water\_feature\_option)

self.visitor\_search\_exhibit\_waterFeature.grid(row=2,column=1)

Label(view, text='Size Min', font="Lucida 12 bold ", width=15).grid(

row=1, column=2, sticky=W)

self.visitor\_search\_exhibit\_size\_min = Entry(view, width=20, textvariable=IntVar(view, value=0))

self.visitor\_search\_exhibit\_size\_min.grid(row=2,column=2)

Label(view, text='Size Max', font="Lucida 12 bold ", width=15).grid(

row=1, column=3, sticky=W)

self.visitor\_search\_exhibit\_size\_max = Entry(view, width=20, textvariable=IntVar(view, value=9000))

self.visitor\_search\_exhibit\_size\_max.grid(row=2,column=3)

Label(view, text='Animal number Min', font="Lucida 12 bold ", width=15).grid(

row=1, column=4, sticky=W)

self.visitor\_search\_animal\_number\_min = Entry(view, width=20, textvariable=IntVar(view, value=0))

self.visitor\_search\_animal\_number\_min.grid(row=2,column=4)

Label(view, text='Animal number Max', font="Lucida 12 bold ", width=15).grid(

row=1, column=5, sticky=W)

self.visitor\_search\_animal\_number\_max = Entry(view, width=20, textvariable=IntVar(view, value=100))

self.visitor\_search\_animal\_number\_max.grid(row=2,column=5)

button = Button(view, text='search', width = 20, command=self.helper\_visitor\_search\_exhibit)

button.grid(row=3,column=0)

def helper\_visitor\_search\_exhibit(self):

temp\_view = Tk()

temp\_view.title('exhibit')

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='Size', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='NumAnimals', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='Water', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

name = self.visitor\_search\_exhibit\_name.get()

waterFeature = self.visitor\_search\_exhibit\_waterFeature\_Entry.get()

size\_min = self.visitor\_search\_exhibit\_size\_min.get()

size\_max = self.visitor\_search\_exhibit\_size\_max.get()

number\_min = self.visitor\_search\_animal\_number\_min.get()

number\_max = self.visitor\_search\_animal\_number\_max.get()

clause = ''

whereClauseStart = False

if name != 'NAME':

if not whereClauseStart:

whereClauseStart = True

clause = "where L.AH = \'%s\'" % (name)

else:

clause += "and L.AH = \'%s\'" % (name)

if waterFeature != '':

if not whereClauseStart:

whereClauseStart = True

clause = "where L.WF = \'%s\'" % (waterFeature)

else:

clause += "and L.WF = \'%s\'" % (waterFeature)

# if clause != '':

if not whereClauseStart:

whereClauseStart = True

clause = "where L.SIZE >= %s and L.SIZE <= %s and L.NumAnimals >= %s and L.NumAnimals <= %s" % (size\_min, size\_max, number\_min, number\_max)

else:

clause += "and L.SIZE >= %s and L.SIZE <= %s and L.NumAnimals >= %s and L.NumAnimals <= %s" % (size\_min, size\_max, number\_min, number\_max)

self.clause = clause;

sql = "SELECT AH, SIZE, NumAnimals, WF from (SELECT animalexhibit as AH, S.size AS SIZE, count(\*) AS NumAnimals, S.waterfeature AS WF from (select animalexhibit, size, waterfeature from (exhibit AS A JOIN animal B) where A.nname = B.animalexhibit) AS S GROUP BY animalexhibit) AS L %s" % (clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

print(row)

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

i += 1

#sorting

i = i + 1

button = Button(temp\_view, text = "name ASC", width=15, command = self.sort\_exhibit\_name\_asc)

button.grid(row=i,column=0, sticky=W)

button = Button(temp\_view, text = "name DESC", width=15, command = self.sort\_exhibit\_name\_desc)

button.grid(row=i + 1,column=0, sticky=W)

button = Button(temp\_view, text = "size ASC", width=15, command = self.sort\_exhibit\_size\_asc)

button.grid(row=i,column=1, sticky=W)

button = Button(temp\_view, text = "size DESC", width=15, command = self.sort\_exhibit\_size\_desc)

button.grid(row=i + 1,column=1, sticky=W)

button = Button(temp\_view, text = "numAnimals ASC", width=15, command = self.sort\_exhibit\_numAnimals\_asc)

button.grid(row=i,column=2, sticky=W)

button = Button(temp\_view, text = "numAnimals DESC", width=15, command = self.sort\_exhibit\_numAnimals\_desc)

button.grid(row=i + 1,column=2, sticky=W)

button = Button(temp\_view, text = "Water ASC", width=15, command = self.sort\_exhibit\_waterFeature\_asc)

button.grid(row=i,column=3, sticky=W)

button = Button(temp\_view, text = "Water DESC", width=15, command = self.sort\_exhibit\_waterFeature\_desc)

button.grid(row=i + 1,column=3, sticky=W)

#exhibit detail

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=i+2, column=0, sticky=W, padx=10)

self.visitor\_helper\_search\_exhibit\_name = Entry(temp\_view, width=15, textvariable=StringVar(temp\_view, value='NAME'))

self.visitor\_helper\_search\_exhibit\_name.grid(row=i+2,column=1)

button = Button(temp\_view, text='Detail', width = 15, command = self.helper\_visitor\_search\_exhibit\_detail)

button.grid(row=i+2,column=2)

def helper\_visitor\_search\_exhibit\_detail(self):

temp\_view = Tk()

temp\_view.title('Exhibit Detail')

name = self.visitor\_helper\_search\_exhibit\_name.get()

clause = "where L.AH = \'%s\'" % (name)

sql = "SELECT AH, SIZE, NumAnimals, WF from (SELECT animalexhibit as AH, S.size AS SIZE, count(\*) AS NumAnimals, S.waterfeature AS WF from (select animalexhibit, size, waterfeature from (exhibit AS A JOIN animal B) where A.nname = B.animalexhibit) AS S GROUP BY animalexhibit) AS L %s" % (clause)

cursor.execute(sql)

temp\_tuple = cursor.fetchone() # holds the exhibit detail

Label(temp\_view, text='Exhibit Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='Size', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='NumAnimals', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='Water', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

Label(temp\_view, text=temp\_tuple[0]).grid(

row=2, column=0, sticky=W, padx=10)

Label(temp\_view, text=temp\_tuple[1]).grid(

row=2, column=1, sticky=W, padx=10)

Label(temp\_view, text=temp\_tuple[2]).grid(

row=2, column=2, sticky=W, padx=10)

Label(temp\_view, text=temp\_tuple[3]).grid(

row=2, column=3, sticky=W, padx=10)

sql = "SELECT nname, species from animal where animalexhibit = \'%s\'" % (name)

self.clause = "where animalexhibit = \'%s\'" % (name)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

Label(temp\_view, text='Animal Name', font="Lucida 12 bold ").grid(

row=3, column=0, sticky=W, padx=10)

Label(temp\_view, text='Species', font="Lucida 12 bold ").grid(

row=3, column=1, sticky=W, padx=10)

i = 4

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

i += 1

#sort

button = Button(temp\_view, text = "Name ASC", width=15, command = self.sort\_visitor\_animal\_name\_asc)

button.grid(row=i,column=0, sticky=W)

button = Button(temp\_view, text = "Name DESC", width=15, command = self.sort\_visitor\_animal\_name\_desc)

button.grid(row=i + 1,column=0, sticky=W)

button = Button(temp\_view, text = "Species ASC", width=15, command = self.sort\_visitor\_animal\_species\_asc)

button.grid(row=i,column=1, sticky=W)

button = Button(temp\_view, text = "Species DESC", width=15, command = self.sort\_visitor\_animal\_species\_desc)

button.grid(row=i + 1,column=1, sticky=W)

#log visit

button = Button(temp\_view, text='Log Visit', width = 20, command=self.helper\_visitor\_search\_exhibit\_logVisit)

button.grid(row=i + 1,column=2)

#animal detail

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=i+2, column=0, sticky=W, padx=10)

self.visitor\_helper\_search\_animal\_name = Entry(temp\_view, width=20, textvariable=StringVar(temp\_view, value='NAME'))

self.visitor\_helper\_search\_animal\_name.grid(row=i+2,column=1)

button = Button(temp\_view, text='Detail', width = 20, command = self.helper\_visitor\_search\_animal\_detail)

button.grid(row=i+2,column=2)

def sort\_visitor\_animal\_name\_asc(self):

print ('animal name asc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

sql = "SELECT nname, species from animal %s ORDER BY nname ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

i += 1

def sort\_visitor\_animal\_name\_desc(self):

print ('animal name desc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

sql = "SELECT nname, species from animal %s ORDER BY nname DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

i += 1

def sort\_visitor\_animal\_species\_asc(self):

print ('animal species asc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

sql = "SELECT nname, species from animal %s ORDER BY species ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

i += 1

def sort\_visitor\_animal\_species\_desc(self):

print ('animal species desc')

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

sql = "SELECT nname, species from animal %s ORDER BY species DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

i += 1

def helper\_visitor\_search\_exhibit\_logVisit(self):

# print (self.user)

sql = "SELECT email from zoo\_user where username = \'%s\'" % (self.user)

cursor.execute(sql)

email = cursor.fetchone()[0]

print (email)

sql\_time = "SELECT NOW()"

cursor.execute(sql\_time)

time = cursor.fetchone()[0]

exhibit = self.visitor\_helper\_search\_exhibit\_name.get()

print (exhibit)

print (time)

sql\_func = "INSERT into exhibitvisits(email, nname, dateandtime) values (\'%s\',\'%s\',\'%s\')" % (email, exhibit, time)

cursor.execute(sql\_func)

db.commit()

# print ("helper\_visitor\_search\_exhibit\_logVisit")

def helper\_visitor\_search\_animal\_detail(self):

temp\_view = Tk()

temp\_view.title('Animal Detail')

name = self.visitor\_helper\_search\_animal\_name.get()

Label(temp\_view, text='name', font="Lucida 12 bold ", width=15).grid(

row=1, column=0)

Label(temp\_view, text='species', font="Lucida 12 bold ", width=15).grid(

row=1, column=1)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ", width=15).grid(

row=1, column=2)

Label(temp\_view, text='age', font="Lucida 12 bold ", width=15).grid(

row=1, column=3)

Label(temp\_view, text='type', font="Lucida 12 bold ", width=15).grid(

row=1, column=4)

sql = "SELECT \* from animal where nname = \'%s\'" % (name)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3)

Label(temp\_view, text=row[4], padx=10).grid(row=i, column=4)

i += 1

##=======================(visitor) search shows========================##

def visitor\_search\_shows(self):

self.view\_visitor\_search\_shows = Tk()

self.view\_visitor\_search\_shows.title('search shows - Visitor')

view = Frame(self.view\_visitor\_search\_shows)

view.pack()

#search

Label(view,text='Name').grid(row=1,column=0, sticky = W)

self.visitor\_view\_shows\_searchname = Entry(view, width=20, textvariable=StringVar(view, value = 'NAME'))

self.visitor\_view\_shows\_searchname.grid(row=2,column=0)

Label(view,text='Exhibit').grid(row=1,column=1, sticky = W)

self.exhibit\_Types1 = {"Jungle","Pacific","Sahara","Mountainous", "Birds", ""}

self.visitor\_view\_shows\_exhibit\_entry = StringVar(view, value='')

self.visitor\_view\_shows\_exhibit = OptionMenu(view, self.visitor\_view\_shows\_exhibit\_entry, \*self.exhibit\_Types1)

self.visitor\_view\_shows\_exhibit.grid(row=2,column=1)

Label(view,text='Date').grid(row=1,column=2, sticky = W)

self.visitor\_view\_shows\_date = Entry(view, width = 20, textvariable=StringVar(view, value = 'YYYY-MM-DD'))

self.visitor\_view\_shows\_date.grid(row=2,column=2)

button = Button(view, text = "search", width=20, command=self.helper\_visitor\_search\_show)

button.grid(row=3,column=0)

def helper\_visitor\_search\_show(self):

temp\_view = Tk()

temp\_view.title("Shows")

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W)

Label(temp\_view, text='date', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W)

#pick a specific one

self.helper\_visitor\_view\_shows\_name = Entry(temp\_view, width=20, textvariable=StringVar(temp\_view, value = 'NAME'))

self.helper\_visitor\_view\_shows\_name.grid(row=2,column=0, sticky=W)

self.helper\_visitor\_view\_shows\_time = Entry(temp\_view, width=20, textvariable=StringVar(temp\_view, value = 'YYYY-MM-DD 00:00:00'))

self.helper\_visitor\_view\_shows\_time.grid(row=2,column=1, sticky=W)

button = Button(temp\_view, text = "Log Visit", width=20, command=self.helper\_visitor\_log\_show)

button.grid(row=2,column=2)

name = self.visitor\_view\_shows\_searchname.get()

exhibit = self.visitor\_view\_shows\_exhibit\_entry.get()

date = self.visitor\_view\_shows\_date.get()

clause = ''

whereClauseStart = False

if name != 'NAME':

if not whereClauseStart:

whereClauseStart = True

clause = "where nname = \'%s\'" %(name)

else:

clause += "and nname = \'%s\'" %(name)

if exhibit != '':

if not whereClauseStart:

whereClauseStart = True

clause = "where showexhibit = \'%s\'" %(exhibit)

else:

clause += "and showexhibit = \'%s\'" %(exhibit)

if date != 'YYYY-MM-DD':

if not whereClauseStart:

whereClauseStart = True

clause = "where showdate = \'%s\'" %(date)

else:

clause += "and showdate = \'%s\'" %(date)

sql = "SELECT nname, showexhibit, dateandtime from zoo\_show %s" % (clause)

self.clause = clause

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 3

for row in review:

Label(temp\_view, text=row[0]).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1]).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2]).grid(row=i, column=2, sticky=W)

i += 1

button = Button(temp\_view, text = "Name ASC", width=15, command = self.sort\_show\_name\_asc)

button.grid(row=i,column=0, sticky=W)

button = Button(temp\_view, text = "Name DESC", width=15, command = self.sort\_show\_name\_desc)

button.grid(row=i + 1,column=0, sticky=W)

button = Button(temp\_view, text = "Exhibit ASC", width=15, command = self.sort\_show\_exhibit\_asc)

button.grid(row=i,column=1, sticky=W)

button = Button(temp\_view, text = "Exhibit DESC", width=15, command = self.sort\_show\_exhibit\_desc)

button.grid(row=i + 1,column=1, sticky=W)

button = Button(temp\_view, text = "Date ASC", width=15, command = self.sort\_show\_time\_asc)

button.grid(row=i,column=2, sticky=W)

button = Button(temp\_view, text = "Date DESC", width=15, command = self.sort\_show\_time\_desc)

button.grid(row=i + 1,column=2, sticky=W)

def helper\_visitor\_log\_show(self):

sql\_email = "SELECT email from visitor where username = \'%s\'" % (self.user)

cursor.execute(sql\_email)

email = cursor.fetchone()[0]

name = self.helper\_visitor\_view\_shows\_name.get()

time = self.helper\_visitor\_view\_shows\_time.get()

sql\_func = "INSERT INTO showvisits(email, nname, dateandtime) values (\'%s\',\'%s\',\'%s\')" % (email, name, time)

cursor.execute(sql\_func)

db.commit()

##=======================(visitor) search for animals==================##

def visitor\_search\_animals(self):

self.view\_visitor\_search\_animals = Tk()

self.view\_visitor\_search\_animals.title('search animals')

view = Frame(self.view\_visitor\_search\_animals)

view.pack()

#animal search

Label(view, text='name', font="Lucida 12 bold ", width=15).grid(

row=1, column=0, sticky=W)

self.visitor\_view\_animals\_name = Entry(view, width=20, textvariable=StringVar(view, value='NAME'))

self.visitor\_view\_animals\_name.grid(row=2,column=0)

Label(view, text='species', font="Lucida 12 bold ", width=15).grid(

row=1, column=1, sticky=W)

self.visitor\_view\_animals\_species = Entry(view, width=20, textvariable=StringVar(view, value='SPECIES'))

self.visitor\_view\_animals\_species.grid(row=2,column=1)

Label(view, text='exhibit', font="Lucida 12 bold ", width=15).grid(

row=1, column=2, sticky=W)

self.exhibit\_Types2 = {'Jungle','Pacific','Sahara','Mountainous', 'Birds', ''}

self.visitor\_view\_animals\_exhibit\_entry = StringVar(view, value='')

self.visitor\_view\_animals\_exhibit = OptionMenu(view, self.visitor\_view\_animals\_exhibit\_entry, \*self.exhibit\_Types2)

self.visitor\_view\_animals\_exhibit.grid(row=2,column=2)

Label(view, text='age', font="Lucida 12 bold ", width=15).grid(

row=1, column=3, sticky=W)

self.visitor\_view\_animal\_age\_min = Entry(view, width=20, textvariable=IntVar(view, value=0))

self.visitor\_view\_animal\_age\_min.grid(row=2,column=3,pady=5)

self.visitor\_view\_animal\_age\_max = Entry(view, width=20, textvariable=IntVar(view, value=100))

self.visitor\_view\_animal\_age\_max.grid(row=3,column=3,pady=5)

Label(view, text='type', font="Lucida 12 bold ", width=15).grid(

row=1, column=4, sticky=W)

self.animal\_types = ['Mammal', 'Bird', 'Amphibian', 'Reptile', 'Fish', 'Invertebrate', '']

self.animal\_types\_entry = StringVar(view, value='')

self.visitor\_view\_animals\_type = OptionMenu(view, self.animal\_types\_entry, \*self.animal\_types)

self.visitor\_view\_animals\_type.grid(row=2,column=4)

button = Button(view, text='search', width = 20, command=self.helper\_visitor\_view\_animals)

button.grid(row=3,column=0)

def helper\_visitor\_view\_animals(self):

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

Label(temp\_view, text='age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W, padx=10)

Label(temp\_view, text='type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W, padx=10)

name = self.visitor\_view\_animals\_name.get()

species = self.visitor\_view\_animals\_species.get()

exhibit = self.visitor\_view\_animals\_exhibit\_entry.get()

min\_age = self.visitor\_view\_animal\_age\_min.get()

max\_age = self.visitor\_view\_animal\_age\_max.get()

animal\_type = self.animal\_types\_entry.get()

clause = ''

whereClauseStart = False

if name != 'NAME':

if not whereClauseStart:

whereClauseStart = True

clause = "where nname = \'%s\'" % (name)

else:

clause += "and nname = \'%s\'" % (name)

if species != 'SPECIES':

if not whereClauseStart:

whereClauseStart = True

clause = "where species = \'%s\'" % (species)

else:

clause += "and species = \'%s\'" % (species)

if animal\_type != '':

if not whereClauseStart:

whereClauseStart = True

clause = "where animal\_type = \'%s\'" % (animal\_type)

else:

clause += "and animal\_type = \'%s\'" % (animal\_type)

if exhibit != '':

if not whereClauseStart:

whereClauseStart = True

clause = "where animalexhibit = \'%s\'" % (exhibit)

else:

clause += "and animalexhibit = \'%s\'" % (exhibit)

if clause == '':

clause = "where age >= %s and age <= %s" % (min\_age, max\_age)

else:

clause += "and age >= %s and age <= %s" % (min\_age, max\_age)

sql = "SELECT nname, species, animalexhibit, age, animal\_type from animal %s" % (clause)

# print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3], padx=10).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=row[4], padx=10).grid(row=i, column=4, sticky=W)

i += 1

# sort

self.clause = clause

button = Button(temp\_view, text = "Name ASC", width=15, command = self.sort\_animal\_name\_asc)

button.grid(row=i,column=0, sticky=W)

button = Button(temp\_view, text = "Name DESC", width=15, command = self.sort\_animal\_name\_desc)

button.grid(row=i + 1,column=0, sticky=W)

button = Button(temp\_view, text = "Species ASC", width=15, command = self.sort\_animal\_species\_asc)

button.grid(row=i,column=1, sticky=W)

button = Button(temp\_view, text = "Species DESC", width=15, command = self.sort\_animal\_species\_desc)

button.grid(row=i + 1,column=1, sticky=W)

button = Button(temp\_view, text = "Exhibit ASC", width=15, command = self.sort\_animal\_exhibit\_asc)

button.grid(row=i,column=2, sticky=W)

button = Button(temp\_view, text = "Exhibit DESC", width=15, command = self.sort\_animal\_exhibit\_desc)

button.grid(row=i + 1,column=2, sticky=W)

button = Button(temp\_view, text = "Age ASC", width=15, command = self.sort\_animal\_age\_asc)

button.grid(row=i,column=3, sticky=W)

button = Button(temp\_view, text = "Age DESC", width=15, command = self.sort\_animal\_age\_desc)

button.grid(row=i + 1,column=3, sticky=W)

button = Button(temp\_view, text = "Type ASC", width=15, command = self.sort\_animal\_type\_asc)

button.grid(row=i,column=4, sticky=W)

button = Button(temp\_view, text = "Type DESC", width=15, command = self.sort\_animal\_type\_desc)

button.grid(row=i + 1,column=4, sticky=W)

##=======================(visitor) view exhibit history================##

def visitor\_view\_exhibit\_history(self):

self.view\_exhibit\_history = Tk()

self.view\_exhibit\_history.title('exhibit history')

view = Frame(self.view\_exhibit\_history)

view.pack()

#animal search

Label(view, text='name', font="Lucida 12 bold ", width=15).grid(

row=1, column=0, sticky=W, padx = 20)

self.visitor\_view\_exhibit\_name = Entry(view, width=20, textvariable=StringVar(view, value='NAME'))

self.visitor\_view\_exhibit\_name.grid(row=2,column=0, padx = 15)

Label(view, text='Time', font="Lucida 12 bold ", width=15).grid(

row=1, column=1, sticky=W, padx = 20)

self.visitor\_view\_exhibit\_time = Entry(view, width=20, textvariable=StringVar(view, value='YYYY-MM-DD 00:00:00'))

self.visitor\_view\_exhibit\_time.grid(row=2,column=1, padx = 15)

Label(view, text='Number of Visits Max', font="Lucida 12 bold ", width=20).grid(

row=1, column=2, sticky=W, padx = 20)

self.visitor\_view\_exhibit\_maxNum\_entry = Entry(view, width=20, textvariable=IntVar(view, value=100))

self.visitor\_view\_exhibit\_maxNum\_entry.grid(row=2,column=2, padx = 15)

Label(view, text='Number of Visits Min', font="Lucida 12 bold ", width=20).grid(

row=1, column=3, sticky=W, padx = 20)

self.visitor\_view\_exhibit\_minNum\_entry = Entry(view, width=20, textvariable=IntVar(view, value=0))

self.visitor\_view\_exhibit\_minNum\_entry.grid(row=2,column=3, padx = 15)

button = Button(view, text='search', width = 20, command=self.helper\_visitor\_exhibit\_history)

button.grid(row=3,column=0, pady = 5)

def helper\_visitor\_exhibit\_history(self):

temp\_view = Tk()

temp\_view.title("History")

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx = 10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx = 10)

Label(temp\_view, text='Number of Visits', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx = 10)

name = self.visitor\_view\_exhibit\_name.get()

time = self.visitor\_view\_exhibit\_time.get()

maxNum = self.visitor\_view\_exhibit\_maxNum\_entry.get()

minNum = self.visitor\_view\_exhibit\_minNum\_entry.get()

clause = 'WHERE email = \'%s\'' % (self.email)

whereClauseStart = False

if name != 'NAME':

clause += "and Anname = \'%s\'" % (name)

if time != 'YYYY-MM-DD 00:00:00':

clause += "and A.dateandtime = \'%s\'" % (time)

if not whereClauseStart:

clause += "and B.count >= %s and B.count <= %s" % (minNum, maxNum)

sql = "SELECT A.nname, A.dateandtime, B.count FROM exhibitvisits A INNER JOIN ((SELECT nname, count(\*) as count FROM exhibitvisits GROUP BY nname)AS B)on A.nname = B.nname %s" % (clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 3

for row in review:

Label(temp\_view, text=row[0]).grid(row=i, column=0, sticky=W, padx = 10)

Label(temp\_view, text=row[1]).grid(row=i, column=1, sticky=W, padx = 10)

Label(temp\_view, text=row[2]).grid(row=i, column=2, sticky=W)

i += 1

# sort

self.clause = clause

button = Button(temp\_view, text = "Name ASC", width=15, command = self.sort\_exhibit\_history\_name\_asc)

button.grid(row=i,column=0, sticky=W)

button = Button(temp\_view, text = "Name DESC", width=15, command = self.sort\_exhibit\_history\_name\_desc)

button.grid(row=i + 1,column=0, sticky=W)

button = Button(temp\_view, text = "Time ASC", width=15, command = self.sort\_exhibit\_history\_time\_asc)

button.grid(row=i,column=1, sticky=W)

button = Button(temp\_view, text = "Time DESC", width=15, command = self.sort\_exhibit\_history\_time\_desc)

button.grid(row=i + 1,column=1, sticky=W)

def sort\_exhibit\_history\_name\_asc(self):

temp\_view = Tk()

temp\_view.title("History")

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx = 10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx = 10)

Label(temp\_view, text='Number of Visits', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx = 10)

sql = "SELECT A.nname, A.dateandtime, B.count FROM exhibitvisits A INNER JOIN ((SELECT nname, count(\*) as count FROM exhibitvisits GROUP BY nname)AS B)on A.nname = B.nname %s order by A.nname asc" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 3

for row in review:

Label(temp\_view, text=row[0]).grid(row=i, column=0, sticky=W, padx = 10)

Label(temp\_view, text=row[1]).grid(row=i, column=1, sticky=W, padx = 10)

Label(temp\_view, text=row[2]).grid(row=i, column=2, sticky=W)

i += 1

def sort\_exhibit\_history\_name\_desc(self):

temp\_view = Tk()

temp\_view.title("History")

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx = 10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx = 10)

Label(temp\_view, text='Number of Visits', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx = 10)

sql = "SELECT A.nname, A.dateandtime, B.count FROM exhibitvisits A INNER JOIN ((SELECT nname, count(\*) as count FROM exhibitvisits GROUP BY nname)AS B)on A.nname = B.nname %s order by A.nname desc" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 3

for row in review:

Label(temp\_view, text=row[0]).grid(row=i, column=0, sticky=W, padx = 10)

Label(temp\_view, text=row[1]).grid(row=i, column=1, sticky=W, padx = 10)

Label(temp\_view, text=row[2]).grid(row=i, column=2, sticky=W)

i += 1

def sort\_exhibit\_history\_time\_asc(self):

temp\_view = Tk()

temp\_view.title("History")

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx = 10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx = 10)

Label(temp\_view, text='Number of Visits', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx = 10)

sql = "SELECT A.nname, A.dateandtime, B.count FROM exhibitvisits A INNER JOIN ((SELECT nname, count(\*) as count FROM exhibitvisits GROUP BY nname)AS B)on A.nname = B.nname %s order by A.dateandtime asc" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 3

for row in review:

Label(temp\_view, text=row[0]).grid(row=i, column=0, sticky=W, padx = 10)

Label(temp\_view, text=row[1]).grid(row=i, column=1, sticky=W, padx = 10)

Label(temp\_view, text=row[2]).grid(row=i, column=2, sticky=W)

i += 1

def sort\_exhibit\_history\_time\_desc(self):

temp\_view = Tk()

temp\_view.title("History")

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx = 10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx = 10)

Label(temp\_view, text='Number of Visits', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx = 10)

sql = "SELECT A.nname, A.dateandtime, B.count FROM exhibitvisits A INNER JOIN ((SELECT nname, count(\*) as count FROM exhibitvisits GROUP BY nname)AS B)on A.nname = B.nname %s order by A.dateandtime desc" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 3

for row in review:

Label(temp\_view, text=row[0]).grid(row=i, column=0, sticky=W, padx = 10)

Label(temp\_view, text=row[1]).grid(row=i, column=1, sticky=W, padx = 10)

Label(temp\_view, text=row[2]).grid(row=i, column=2, sticky=W)

i += 1

##=======================(visitor) view show history===================##

def visitor\_view\_show\_history(self):

self.view\_show\_history = Tk()

self.view\_show\_history.title('show history')

view = Frame(self.view\_show\_history)

view.pack()

#search

Label(view,text='Name', width = 20).grid(row=1,column=0, sticky = W, padx = 5, pady = 10)

self.visitor\_view\_shows\_searchname = Entry(view, width=20, textvariable=StringVar(view, value = 'NAME'))

self.visitor\_view\_shows\_searchname.grid(row=2,column=0)

Label(view,text='Exhibit', width = 20).grid(row=1,column=1, sticky = W, padx = 5, pady = 10)

self.exhibit\_Types1 = {"Jungle","Pacific","Sahara","Mountainous", "Birds", ""}

self.visitor\_view\_shows\_exhibit\_entry = StringVar(view, value='')

self.visitor\_view\_shows\_exhibit = OptionMenu(view, self.visitor\_view\_shows\_exhibit\_entry, \*self.exhibit\_Types1)

self.visitor\_view\_shows\_exhibit.grid(row=2,column=1)

Label(view,text='Date', width = 20).grid(row=1,column=2, sticky = W, padx = 5, pady = 10)

self.visitor\_view\_shows\_date = Entry(view, width = 20, textvariable=StringVar(view, value = 'YYYY-MM-DD'))

self.visitor\_view\_shows\_date.grid(row=2,column=2)

button = Button(view, text = "search", width=20, command=self.helper\_visitor\_view\_shows\_search)

button.grid(row=3,column=0, padx = 5, pady = 10)

def helper\_visitor\_view\_shows\_search(self):

temp\_view = Tk()

temp\_view.title("Shows")

Label(temp\_view, text='name', font="Lucida 12 bold ", width = 20).grid(

row=1, column=0, sticky=W, padx = 5)

Label(temp\_view, text='date', font="Lucida 12 bold ", width = 20).grid(

row=1, column=1, sticky=W, padx = 5)

Label(temp\_view, text='exhibit', font="Lucida 12 bold ", width = 20).grid(

row=1, column=2, sticky=W, padx = 5)

name = self.visitor\_view\_shows\_searchname.get()

exhibit = self.visitor\_view\_shows\_exhibit\_entry.get()

date = self.visitor\_view\_shows\_date.get()

clause = 'WHERE A.nname = B.nname and A.email = \'%s\'' % (self.email)

if name != 'NAME':

clause += "and B.nname = \'%s\'" %(name)

if exhibit != '':

clause += "and B.showexhibit = \'%s\'" %(exhibit)

if date != 'YYYY-MM-DD':

clause += "and B.showdate = \'%s\'" %(date)

sql = "SELECT B.nname, A.dateandtime, B.showexhibit FROM (showvisits A JOIN zoo\_show B) %s" % (clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], width = 20).grid(row=i, column=0, sticky=W, padx = 2, pady = 5)

Label(temp\_view, text=row[1], width = 20).grid(row=i, column=1, sticky=W, padx = 2, pady = 5)

Label(temp\_view, text=row[2], width = 20).grid(row=i, column=2, sticky=W, padx = 2, pady = 5)

i += 1

# sort

self.clause = clause

button = Button(temp\_view, text = "Name ASC", width=15, command = self.sort\_show\_history\_name\_asc)

button.grid(row=i,column=0, sticky=W)

button = Button(temp\_view, text = "Name DESC", width=15, command = self.sort\_show\_history\_name\_desc)

button.grid(row=i + 1,column=0, sticky=W)

button = Button(temp\_view, text = "Time ASC", width=15, command = self.sort\_show\_history\_time\_asc)

button.grid(row=i,column=1, sticky=W)

button = Button(temp\_view, text = "Time DESC", width=15, command = self.sort\_show\_history\_time\_desc)

button.grid(row=i + 1,column=1, sticky=W)

def sort\_show\_history\_name\_asc(self):

temp\_view = Tk()

temp\_view.title("History")

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx = 10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx = 10)

Label(temp\_view, text='Number of Visits', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx = 10)

sql = "SELECT B.nname, A.dateandtime, B.showexhibit FROM (showvisits A JOIN zoo\_show B) %s ORDER BY B.nname asc" % (self.clause)

# sql = "SELECT nname, showexhibit, dateandtime from zoo\_show %s ORDER BY nname ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 3

for row in review:

Label(temp\_view, text=row[0]).grid(row=i, column=0, sticky=W, padx = 10)

Label(temp\_view, text=row[1]).grid(row=i, column=1, sticky=W, padx = 10)

Label(temp\_view, text=row[2]).grid(row=i, column=2, sticky=W)

i += 1

def sort\_show\_history\_name\_desc(self):

temp\_view = Tk()

temp\_view.title("History")

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx = 10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx = 10)

Label(temp\_view, text='Number of Visits', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx = 10)

sql = "SELECT B.nname, A.dateandtime, B.showexhibit FROM (showvisits A JOIN zoo\_show B) %s ORDER BY B.nname desc" % (self.clause)

# sql = "SELECT nname, showexhibit, dateandtime from zoo\_show %s ORDER BY nname DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 3

for row in review:

Label(temp\_view, text=row[0]).grid(row=i, column=0, sticky=W, padx = 10)

Label(temp\_view, text=row[1]).grid(row=i, column=1, sticky=W, padx = 10)

Label(temp\_view, text=row[2]).grid(row=i, column=2, sticky=W)

i += 1

def sort\_show\_history\_time\_asc(self):

temp\_view = Tk()

temp\_view.title("History")

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx = 10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx = 10)

Label(temp\_view, text='Number of Visits', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx = 10)

sql = "SELECT B.nname, A.dateandtime, B.showexhibit FROM (showvisits A JOIN zoo\_show B) %s ORDER BY A.dateandtime asc" % (self.clause)

# sql = "SELECT nname, showexhibit, dateandtime from zoo\_show %s ORDER BY dateandtime ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 3

for row in review:

Label(temp\_view, text=row[0]).grid(row=i, column=0, sticky=W, padx = 10)

Label(temp\_view, text=row[1]).grid(row=i, column=1, sticky=W, padx = 10)

Label(temp\_view, text=row[2]).grid(row=i, column=2, sticky=W)

i += 1

def sort\_show\_history\_time\_desc(self):

temp\_view = Tk()

temp\_view.title("History")

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx = 10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx = 10)

Label(temp\_view, text='Number of Visits', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx = 10)

# sql = "SELECT nname, showexhibit, dateandtime from zoo\_show %s ORDER BY dateandtime DESC" % (self.clause)

sql = "SELECT B.nname, A.dateandtime, B.showexhibit FROM (showvisits A JOIN zoo\_show B) %s ORDER BY A.dateandtime desc" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 3

for row in review:

Label(temp\_view, text=row[0]).grid(row=i, column=0, sticky=W, padx = 10)

Label(temp\_view, text=row[1]).grid(row=i, column=1, sticky=W, padx = 10)

Label(temp\_view, text=row[2]).grid(row=i, column=2, sticky=W)

i += 1

##=======================(visitor) log out=============================##

def visitor\_log\_out(self):

self.main.destroy()

#========================================================================

# staff

#========================================================================

##=======================(staff) log out===============================##

def staff\_log\_out(self):

self.main.destroy()

##=======================(staff) view shows============================##

def staff\_view\_shows(self):

self.staff\_view\_shows = Tk()

self.staff\_view\_shows.title('View Shows')

view\_show\_frame = Frame(self.staff\_view\_shows)

view\_show\_frame.pack()

# backend code

sql = "SELECT nname, showexhibit, dateandtime FROM zoo\_show where showstaff = \'%s\'" % (self.email)

cursor.execute(sql)

# reivew = cursor.fetchall()

review = []

for row in cursor:

review.append(row)

# insert chart headers

Label(view\_show\_frame, text='Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W)

Label(view\_show\_frame, text='Exhibit', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W)

Label(view\_show\_frame, text='Time', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W)

i = 2

for row in review:

Label(view\_show\_frame, text=row[0]).grid(row=i, column=0, sticky=W)

Label(view\_show\_frame, text=row[1]).grid(row=i, column=1, sticky=W)

Label(view\_show\_frame, text=row[2]).grid(row=i, column=2, sticky=W)

i += 1

#sort

self.clause = "where showstaff = \'%s\'" % (self.email)

button = Button(view\_show\_frame, text = "Name ASC", width=15, command = self.sort\_show\_name\_asc)

button.grid(row=i,column=0, sticky=W)

button = Button(view\_show\_frame, text = "Name DESC", width=15, command = self.sort\_show\_name\_desc)

button.grid(row=i + 1,column=0, sticky=W)

button = Button(view\_show\_frame, text = "Exhibit ASC", width=15, command = self.sort\_show\_exhibit\_asc)

button.grid(row=i,column=1, sticky=W)

button = Button(view\_show\_frame, text = "Exhibit DESC", width=15, command = self.sort\_show\_exhibit\_desc)

button.grid(row=i + 1,column=1, sticky=W)

button = Button(view\_show\_frame, text = "Date ASC", width=15, command = self.sort\_show\_time\_asc)

button.grid(row=i,column=2, sticky=W)

button = Button(view\_show\_frame, text = "Date DESC", width=15, command = self.sort\_show\_time\_desc)

button.grid(row=i + 1,column=2, sticky=W)

##=======================(staff) search animals========================##

def staff\_search\_animals(self):

self.staff\_search\_animals = Tk()

self.staff\_search\_animals.title('Search animals')

view = Frame(self.staff\_search\_animals)

view.pack()

#search

Label(view,text='Name').grid(row=1,column=0, sticky = W)

self.staff\_view\_shows\_searchname = Entry(view, width=20, textvariable=StringVar(view, value = 'NAME'))

self.staff\_view\_shows\_searchname.grid(row=2,column=0)

Label(view,text='Species').grid(row=1,column=1, sticky = W)

self.staff\_view\_shows\_searchspecies = Entry(view, width=20, textvariable=StringVar(view, value = 'SPECIES'))

self.staff\_view\_shows\_searchspecies.grid(row=2,column=1)

Label(view,text='Exhibit').grid(row=1,column=2, sticky = W)

self.exhibit\_Types1 = {'Jungle','Pacific','Sahara','Mountainous', 'Birds', ''}

self.staff\_view\_exhibit\_entry = StringVar(view,value='')

self.staff\_view\_exhibit = OptionMenu(view, self.staff\_view\_exhibit\_entry, \*self.exhibit\_Types1)

self.staff\_view\_exhibit.grid(row=2,column=2)

Label(view,text='Age').grid(row=1, column=3, sticky = W)

self.min\_age\_entry = Entry(view, width=10, textvariable=IntVar(view, value = 0))

self.min\_age\_entry.grid(row=2, column=3)

self.max\_age\_entry = Entry(view, width=10, textvariable=IntVar(view, value = 100))

self.max\_age\_entry.grid(row=3, column=3)

Label(view,text='Type').grid(row=1,column=4, sticky = W)

self.animal\_Types1 = {'Fish','Bird','Reptile','Mammal', 'Amphibian', 'Invertebrate', ''}

self.staff\_view\_animal\_entry = StringVar(view, value='')

self.staff\_view\_animal = OptionMenu(view, self.staff\_view\_animal\_entry, \*self.animal\_Types1)

self.staff\_view\_animal.grid(row=2,column=4)

button = Button(view, text = "search", width=20, command=self.helper\_staff\_search\_animal)

button.grid(row=4,column=0)

def helper\_staff\_search\_animal(self):

temp\_view = Tk()

temp\_view.title("Animals")

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W)

Label(temp\_view, text='Species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W)

Label(temp\_view, text='Exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W)

Label(temp\_view, text='Age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W)

Label(temp\_view, text='Type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W)

name = self.staff\_view\_shows\_searchname.get()

species = self.staff\_view\_shows\_searchspecies.get()

animalexhibit = self.staff\_view\_exhibit\_entry.get()

min\_age = self.min\_age\_entry.get()

max\_age = self.max\_age\_entry.get()

animal\_type = self.staff\_view\_animal\_entry.get()

clause = ''

whereClauseStart = False

if name != 'NAME':

if not whereClauseStart:

whereClauseStart = True

clause = "where nname = \'%s\'" %(name)

else:

clause += "and nname = \'%s\'" %(name)

if species != 'SPECIES':

if not whereClauseStart:

whereClauseStart = True

clause = "where species = \'%s\'" %(species)

else:

clause += "and species = \'%s\'" %(species)

if animal\_type != '':

if not whereClauseStart:

whereClauseStart = True

clause = "where animal\_type = \'%s\'" %(animal\_type)

else:

clause += "and animal\_type = \'%s\'" %(animal\_type)

if animalexhibit != '':

if not whereClauseStart:

whereClauseStart = True

clause = "where animalexhibit = \'%s\'" %(animalexhibit)

else:

clause += "and animalexhibit = \'%s\'" %(animalexhibit)

if clause == '':

clause = "where age >= %s and age <= %s" % (min\_age, max\_age)

else:

clause += "and age >= %s and age <= %s" % (min\_age, max\_age)

sql = "SELECT nname, species, animalexhibit, age, animal\_type FROM animal %s" % (clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0]).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1]).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2]).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=row[3]).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=row[4]).grid(row=i, column=4, sticky=W)

i += 1

# sort

self.clause = clause

button = Button(temp\_view, text = "Name ASC", width=15, command = self.sort\_animal\_name\_asc)

button.grid(row=i,column=0, sticky=W)

button = Button(temp\_view, text = "Name DESC", width=15, command = self.sort\_animal\_name\_desc)

button.grid(row=i + 1,column=0, sticky=W)

button = Button(temp\_view, text = "Species ASC", width=15, command = self.sort\_animal\_species\_asc)

button.grid(row=i,column=1, sticky=W)

button = Button(temp\_view, text = "Species DESC", width=15, command = self.sort\_animal\_species\_desc)

button.grid(row=i + 1,column=1, sticky=W)

button = Button(temp\_view, text = "Exhibit ASC", width=15, command = self.sort\_animal\_exhibit\_asc)

button.grid(row=i,column=2, sticky=W)

button = Button(temp\_view, text = "Exhibit DESC", width=15, command = self.sort\_animal\_exhibit\_desc)

button.grid(row=i + 1,column=2, sticky=W)

button = Button(temp\_view, text = "Age ASC", width=15, command = self.sort\_animal\_age\_asc)

button.grid(row=i,column=3, sticky=W)

button = Button(temp\_view, text = "Age DESC", width=15, command = self.sort\_animal\_age\_desc)

button.grid(row=i + 1,column=3, sticky=W)

button = Button(temp\_view, text = "Type ASC", width=15, command = self.sort\_animal\_type\_asc)

button.grid(row=i,column=4, sticky=W)

button = Button(temp\_view, text = "Type DESC", width=15, command = self.sort\_animal\_type\_desc)

button.grid(row=i + 1,column=4, sticky=W)

# animal detail

self.staff\_view\_animal\_name = Entry(temp\_view, width=20, textvariable=StringVar(temp\_view, value = 'NAME'))

self.staff\_view\_animal\_name.grid(row=i + 2,column=0)

self.staff\_view\_animal\_species = Entry(temp\_view, width=20, textvariable=StringVar(temp\_view, value = 'SPECIES'))

self.staff\_view\_animal\_species.grid(row=i + 2,column=1)

button = Button(temp\_view, text = "Animal Care", width=15, command = self.helper\_staff\_animal\_care)

button.grid(row=i + 2,column=2, sticky=W)

def helper\_staff\_animal\_care(self):

self.view\_helper\_staff\_animal\_care = Tk()

self.view\_helper\_staff\_animal\_care.title('Animal Care')

temp\_view = Frame(self.view\_helper\_staff\_animal\_care)

temp\_view.pack()

Label(temp\_view, text='Name', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W)

Label(temp\_view, text='Species', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W)

Label(temp\_view, text='Exhibit', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W)

Label(temp\_view, text='Age', font="Lucida 12 bold ").grid(

row=1, column=3, sticky=W)

Label(temp\_view, text='Type', font="Lucida 12 bold ").grid(

row=1, column=4, sticky=W)

name = self.staff\_view\_animal\_name.get()

species = self.staff\_view\_animal\_species.get()

sql = "SELECT nname, species, animalexhibit, age, animal\_type FROM animal where nname = \'%s\' and species = \'%s\'" % (name, species)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

Label(temp\_view, text=review[0][0]).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=review[0][1]).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=review[0][2]).grid(row=i, column=2, sticky=W)

Label(temp\_view, text=review[0][3]).grid(row=i, column=3, sticky=W)

Label(temp\_view, text=review[0][4]).grid(row=i, column=4, sticky=W)

# animal care history

Label(temp\_view, text='Staff', font="Lucida 12 bold ").grid(

row=i + 1, column=0, sticky=W)

Label(temp\_view, text='Note', font="Lucida 12 bold ").grid(

row=i + 1, column=1, sticky=W)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=i + 1, column=2, sticky=W)

sql = "SELECT email, notetext, notetime FROM note WHERE nname = \'%s\'" % (name)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

a = i + 2

for row in review:

Label(temp\_view, text=row[0]).grid(row=a, column=0, sticky=W)

Label(temp\_view, text=row[1]).grid(row=a, column=1, sticky=W)

Label(temp\_view, text=row[2]).grid(row=a, column=2, sticky=W)

a += 1

# sort

self.clause = "WHERE nname = \'%s\'" % (name)

button = Button(temp\_view, text = "Staff ASC", width=15, command = self.sort\_animal\_care\_staff\_asc)

button.grid(row=a,column=0, sticky=W)

button = Button(temp\_view, text = "Staff DESC", width=15, command = self.sort\_animal\_care\_staff\_desc)

button.grid(row=a + 1,column=0, sticky=W)

button = Button(temp\_view, text = "Note ASC", width=15, command = self.sort\_animal\_care\_note\_asc)

button.grid(row=a,column=1, sticky=W)

button = Button(temp\_view, text = "Note DESC", width=15, command = self.sort\_animal\_care\_note\_desc)

button.grid(row=a + 1,column=1, sticky=W)

button = Button(temp\_view, text = "Time ASC", width=15, command = self.sort\_animal\_care\_time\_asc)

button.grid(row=a,column=2, sticky=W)

button = Button(temp\_view, text = "Time DESC", width=15, command = self.sort\_animal\_care\_time\_desc)

button.grid(row=a + 1,column=2, sticky=W)

Label(temp\_view, text='Note', font="Lucida 12 bold ").grid(

row=a + 2, column=4, sticky=W)

self.note = Entry(temp\_view, width=100, textvariable=StringVar(temp\_view, value = ''))

self.note.grid(row=a + 2,column=5)

button = Button(temp\_view, text = "Log", width=15, command = self.helper\_staff\_log\_animal\_care)

button.grid(row=a + 2,column=0, sticky=W)

def sort\_animal\_care\_staff\_asc(self):

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='Staff', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='Note', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

sql = "SELECT email, notetext, notetime FROM note %s ORDER BY email ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

i += 1

def sort\_animal\_care\_staff\_desc(self):

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='Staff', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='Note', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

sql = "SELECT email, notetext, notetime FROM note %s ORDER BY email DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

i += 1

def sort\_animal\_care\_note\_asc(self):

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='Staff', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='Note', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

sql = "SELECT email, notetext, notetime FROM note %s ORDER BY notetext ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

i += 1

def sort\_animal\_care\_note\_desc(self):

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='Staff', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='Note', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

sql = "SELECT email, notetext, notetime FROM note %s ORDER BY notetext DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

i += 1

def sort\_animal\_care\_time\_asc(self):

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='Staff', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='Note', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

sql = "SELECT email, notetext, notetime FROM note %s ORDER BY notetime ASC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

i += 1

def sort\_animal\_care\_time\_desc(self):

temp\_view = Tk()

temp\_view.title('Animals')

Label(temp\_view, text='Staff', font="Lucida 12 bold ").grid(

row=1, column=0, sticky=W, padx=10)

Label(temp\_view, text='Note', font="Lucida 12 bold ").grid(

row=1, column=1, sticky=W, padx=10)

Label(temp\_view, text='Time', font="Lucida 12 bold ").grid(

row=1, column=2, sticky=W, padx=10)

sql = "SELECT email, notetext, notetime FROM note %s ORDER BY notetime DESC" % (self.clause)

print (sql)

cursor.execute(sql)

review = []

for row in cursor:

review.append(row)

i = 2

for row in review:

Label(temp\_view, text=row[0], padx=10).grid(row=i, column=0, sticky=W)

Label(temp\_view, text=row[1], padx=10).grid(row=i, column=1, sticky=W)

Label(temp\_view, text=row[2], padx=10).grid(row=i, column=2, sticky=W)

i += 1

def helper\_staff\_log\_animal\_care(self):

name = self.staff\_view\_animal\_name.get()

species = self.staff\_view\_animal\_species.get()

email = self.email

text = self.note.get()

sql\_time = "SELECT NOW()"

cursor.execute(sql\_time)

time = cursor.fetchone()[0]

sql = "INSERT into note(nname, species, email, notetime, notetext) VALUES (\'%s\',\'%s\',\'%s\',\'%s\',\'%s\')" % (name, species, email, time, text)

print(sql)

cursor.execute(sql)

db.commit()

self.view\_helper\_staff\_animal\_care.destroy()

a = GUI()

cursor.close()

db.close()