Objectives:

* TKINTER and JOINING Tables.

**There are 2 challenge exercises, each worth 50%.**

**Project #1 (JOINING TABLES).**

Create the following **Orders** Table that will reference the **Customers** table created earlier.

A close-up of words

Description automatically generated

Insert the following data using the insert statement.

A close up of black text

Description automatically generated

Joining the Customers and Orders Tables.

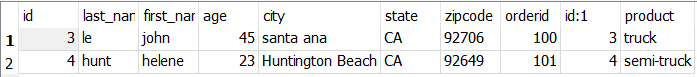
A black and white text

Description automatically generated

TKINTER DISPLAYING JOINED TABLES.

**Challenge Exercise 1:** Your task is to create a button that will display the JOIN statement (Above) in a message box.

**#1 Print screen the running application below here. (NO CODE NEEDED).**

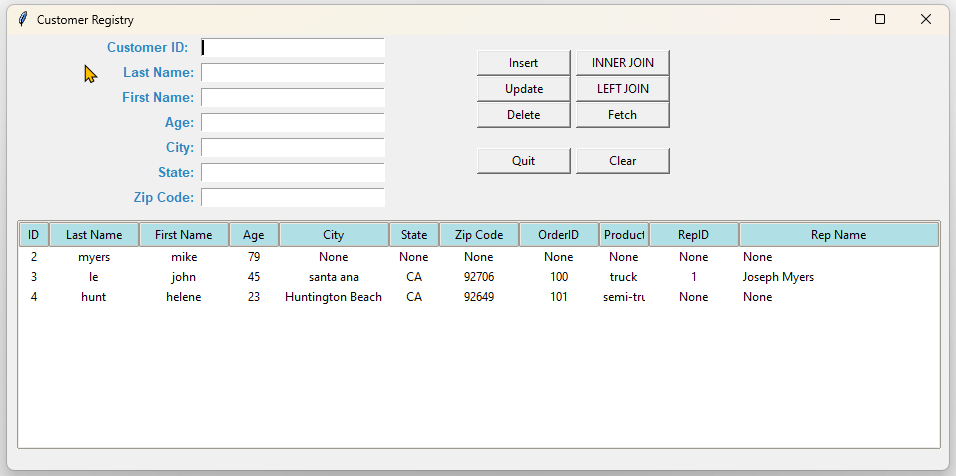


Note: I changed some of the values and added new entry into the Orders table

**Challenge Exercise 2:** Your second task is to create a 3rd table named *Representative* with the following columns: RepID, ID (this column will act as the foreign key that will reference the ID column for the Customers table), and Rep\_Name.

Then enter the data of your choice. Finally, join the Customers, Orders, and Representative tables, and display the data from TKINTER.

**#2 Print screen the running application below here. (NO CODE NEEDED).**

****

**Submit this document to Module 5 Class Exercise.**

Code:

import tkinter as tk

from tkinter import ttk

import sqlite3 as sql

from tkinter import messagebox

#import sqlserver as ss

def insert\_data():

last\_name = txtln.get()

first\_name = txtfn.get()

age = txtage.get()

city = txtcity.get()

state = txtstate.get()

zip = txtzip.get()

#open connection

conn = sql.connect("C:\\Users\\17147\\Desktop\\SQLLite\\PythonClassDB.db")

crs = conn.cursor()

crs.execute("INSERT INTO customers (last\_name, first\_name, age, city, state, zipcode) VALUES(?,?,?,?,?,?)", (last\_name, first\_name, age, city, state, zip))

conn.commit() #Commit transaction

conn.close() #Close connection

messagebox.showinfo(title="DB Action", message="Insert successful.")

txtid.delete(0, tk.END)

txtln.delete(0, tk.END)

txtfn.delete(0, tk.END)

txtage.delete(0, tk.END)

txtcity.delete(0, tk.END)

txtstate.delete(0, tk.END)

txtzip.delete(0, tk.END)

txtid.focus\_set()

def update\_data():

id = txtid.get()

last\_name = txtln.get()

first\_name = txtfn.get()

age = txtage.get()

city = txtcity.get()

state = txtstate.get()

zip = txtzip.get()

conn = sql.connect("C:\\Users\\17147\\Desktop\\SQLLite\\PythonClassDB.db")

crs = conn.cursor()

crs.execute("UPDATE customers SET last\_name=?, first\_name=?, age=?, city=?, state=?, zipcode=? WHERE id=?", (last\_name, first\_name, age, city, state, zip, id))

conn.commit()

conn.close

messagebox.showinfo(title="DB Action", message=f"Customer ID {id} was updated successful.")

txtid.delete(0, tk.END)

txtln.delete(0, tk.END)

txtfn.delete(0, tk.END)

txtage.delete(0, tk.END)

txtcity.delete(0, tk.END)

txtstate.delete(0, tk.END)

txtzip.delete(0, tk.END)

txtid.focus\_set()

def delete\_data():

id = txtid.get()

conn = sql.connect("C:\\Users\\17147\\Desktop\\SQLLite\\PythonClassDB.db")

crs = conn.cursor()

crs.execute("DELETE FROM customers WHERE id=?", (id))

conn.commit()

conn.close

messagebox.showinfo(title="DB Action", message=f"Customer ID {id} was deleted successful.")

txtid.delete(0, tk.END)

txtln.delete(0, tk.END)

txtfn.delete(0, tk.END)

txtage.delete(0, tk.END)

txtcity.delete(0, tk.END)

txtstate.delete(0, tk.END)

txtzip.delete(0, tk.END)

txtid.focus\_set()

def fetch\_data():

conn = sql.connect("C:\\Users\\17147\\Desktop\\SQLLite\\PythonClassDB.db")

crs = conn.cursor()

crs.execute("SELECT \* FROM customers")

rows = crs.fetchall()

for row in rows:

#print(row)

tree.insert("", tk.END, values=row)

conn.close()

def fetch\_joindata1():

conn = sql.connect("C:\\Users\\17147\\Desktop\\SQLLite\\PythonClassDB.db")

crs = conn.cursor()

sqlstatement = "SELECT \* FROM customers c INNER JOIN orders o ON c.id = o.id"

crs.execute(sqlstatement)

rows = crs.fetchall()

for row in rows:

#print(row)

tree.insert("", tk.END, values=row)

conn.close()

def fetch\_joindata2():

conn = sql.connect("C:\\Users\\17147\\Desktop\\SQLLite\\PythonClassDB.db")

crs = conn.cursor()

sqlstatement = """SELECT c.id, c.last\_name, c.first\_name, c.age, c.city, c.state, c.zipcode,

o.orderid, o.product, r.repid, r.rep\_name

FROM customers c

LEFT JOIN orders o ON c.id = o.id

LEFT JOIN representatives r ON c.id = r.id"""

crs.execute(sqlstatement)

rows = crs.fetchall()

for row in rows:

#print(row)

tree.insert("", tk.END, values=row)

conn.close()

def quit():

win.quit()

win.destroy()

def cleartext():

txtid.delete(0, tk.END)

txtfn.delete(0, tk.END)

txtln.delete(0, tk.END)

txtage.delete(0, tk.END)

txtcity.delete(0, tk.END)

txtstate.delete(0, tk.END)

txtzip.delete(0, tk.END)

txtid.focus\_set()

for item in tree.get\_children():

tree.delete(item)

#Set up window form

win = tk.Tk()

win.title('Customer Registry')

win.geometry("942x435")

# Input

lblid = tk.Label(win, text = "Customer ID: ", font="Arial 10 bold", width=12, fg="#2E86C1")

lblid.grid(column=0, row=1, sticky="e") #Label widge

id = tk.StringVar() #Manage the Entry widget

txtid = tk.Entry(win, width=30, textvariable=id)

txtid.grid(column=1,row=1, ipadx = 0, padx = 0, pady=3, sticky="w")

txtid.focus\_set()

lblln = tk.Label(win, text = "Last Name: ", font="Arial 10 bold", width=12, anchor="e", fg="#2E86C1")

lblln.grid(column=0, row=2, sticky="e") #Label widge

ln = tk.StringVar() #Manage the Entry widget

txtln = tk.Entry(win, width=30, textvariable=ln)

txtln.grid(column=1,row=2, ipadx = 0, padx = 0, pady=3, sticky="w")

lblfn = tk.Label(win, text = "First Name: ", font="Arial 10 bold", width=12, anchor="e", fg="#2E86C1")

lblfn.grid(column=0, row=3, sticky="e") #Label widge

fn = tk.StringVar() #Manage the Entry widget

txtfn = tk.Entry(win, width=30, textvariable=fn)

txtfn.grid(column=1,row=3, ipadx = 0, padx = 0, pady=3, sticky="w")

lblage = tk.Label(win, text = "Age: ", font="Arial 10 bold", width=12, anchor="e", fg="#2E86C1")

lblage.grid(column=0, row=4, sticky="e") #Label widge

age = tk.StringVar() #Manage the Entry widget

txtage = tk.Entry(win, width=30, textvariable=age)

txtage.grid(column=1, row=4, ipadx = 0, padx = 0, pady=3, sticky="w")

lblcity = tk.Label(win, text = "City: ", font="Arial 10 bold", width=12, anchor="e", fg="#2E86C1")

lblcity.grid(column=0, row=5, sticky="e") #Label widge

city = tk.StringVar() #Manage the Entry widget

txtcity = tk.Entry(win, width=30, textvariable=city)

txtcity.grid(column=1, row=5, ipadx = 0, padx = 0, pady=3, sticky="w")

lblstate = tk.Label(win, text = "State: ", font="Arial 10 bold", width=12, anchor="e", fg="#2E86C1")

lblstate.grid(column=0, row=6, sticky="e") #Label widge

state = tk.StringVar() #Manage the Entry widget

txtstate = tk.Entry(win, width=30, textvariable=state)

txtstate.grid(column=1, row=6, ipadx = 0, padx = 0, pady=3, sticky="w")

lblzip = tk.Label(win, text = "Zip Code: ", font="Arial 10 bold", width=12, anchor="e", fg="#2E86C1")

lblzip.grid(column=0, row=7, sticky="e") #Label widge

zip = tk.StringVar() #Manage the Entry widget

txtzip = tk.Entry(win, width=30, textvariable=zip)

txtzip.grid(column=1, row=7, ipadx = 0, padx = 0, pady=3, sticky="w")

# Create buttons on windows form

frm3 = tk.Frame(win)

frm3.grid(column=2, row=1, padx = 0, ipadx = 0, rowspan=8, sticky="w", columns=1)

btninsert = tk.Button(frm3, text="Insert", command=insert\_data, width=12)

btninsert.grid(column=0, row=0, padx=0, ipadx=0)

btnupdate = tk.Button(frm3, text="Update", command=update\_data, width=12)

btnupdate.grid(column=0, row=1, padx=0, ipadx=0)

btndelete = tk.Button(frm3, text="Delete", command=delete\_data, width=12)

btndelete.grid(column=0, row=2, padx=0, ipadx=0)

btnjoin1 = tk.Button(frm3, text="INNER JOIN", command=fetch\_joindata1, width=12)

btnjoin1.grid(column=1, row=0, padx=5)

btnjoin2 = tk.Button(frm3, text="LEFT JOIN", command=fetch\_joindata2, width=12)

btnjoin2.grid(column=1, row=1, padx=5)

btnfetch = tk.Button(frm3, text="Fetch", command=fetch\_data, width=12)

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

btnquit = tk.Button(frm3, text="Quit", command=quit, width=12)

btnquit.grid(column=0, row=5, pady=20)

btnclear = tk.Button(frm3, text="Clear", command=cleartext, width=12)

btnclear.grid(column=1, row=5, pady=20)

#Tree view of data

columns = ("id", "last\_name", "first\_name", "age", "city", "state", "zip", "orderid", "product", "repid", "rep\_name")

frm5 = tk.Frame(win)

frm5.grid(column=0, row=9, columnspan=5, padx=10, pady=10)

tree = ttk.Treeview(frm5, column=columns, show='headings', height=10)

style = ttk.Style()

style.theme\_use('clam')

style.configure("Treeview", background="white", fieldbackground="white", foreground="black")

style.configure('Treeview.Heading', background="PowderBlue")

tree.column("#1", anchor=tk.CENTER, width=30)

tree.heading("id", text="ID")

tree.column("#2", anchor=tk.CENTER, width=90)

tree.heading("last\_name", text="Last Name")

tree.column("#3", anchor=tk.CENTER, width=90)

tree.heading("first\_name", text="First Name")

tree.column("#4", anchor=tk.CENTER, width=50)

tree.heading("age", text="Age")

tree.column("#5", anchor=tk.CENTER, width=110)

tree.heading("city", text="City")

tree.column("#6", anchor=tk.CENTER, width=50)

tree.heading("state", text="State")

tree.column("#7", anchor=tk.CENTER, width=80)

tree.heading("zip", text="Zip Code")

tree.column("#8", anchor=tk.CENTER, width=80)

tree.heading("orderid", text="OrderID")

tree.column("#9", anchor=tk.CENTER, width=90)

tree.heading("product", text="Product")

tree.column("#9", anchor=tk.CENTER, width=50)

tree.heading("repid", text="RepID")

tree.column("#10", anchor=tk.CENTER, width=90)

tree.heading("rep\_name", text="Rep Name")

tree.grid(column=0, row=0, ipadx = 0, padx = 0, columnspan=3)

win.mainloop()