Python Programming

Day 16: Introduction to Database (2)

Introduction to Database

Insert	Delete
Select	Update

Color and symbol meaning

Hint

Preferred

Student's activity

Practice code

	Keyword
	In-built functions
	Strings
	Output

AGGREGATE Functions

Aggregate functions are used to compute against a "returned column of numeric data" from your **SELECT** statement. They basically summarize the results of a particular column of selected data.

- **AVG**
- **COUNT**
- **SUM**
- **MIN**
- **MAX**

SELECT COUNT Statement

The **COUNT()** function returns the number of rows that matches a specified criteria.

Syntax

SELECT COUNT(<column_name>)

FROM <table_name>

Sample

SELECT COUNT(*)
FROM items_ordered

items_ordered

customerid	order_date	item	quantity	price
10330	30-Jun-1999	Pogo stick	1	28.00
10101	30-Jun-1999	Raft	1	58.00
10298	01-Jul-1999	Skateboard	1	33.00
10101	01-Jul-1999	Life Vest	4	125.00
10299	06-Jul-1999	Parachute	1	1250.00
10339	27-Jul-1999	Umbrella	1	4.50
10449	13-Aug-1999	Unicycle	1	180.79
10439	14-Aug-1999	Ski Poles	2	25.50
10101	18-Aug-1999	Rain Coat	1	18.30

When attribute is replaced with * the query returns number of rows in the table. Example:

SELECT SUM Statement

Sample

The **SUM()** function returns the sum of the numeric values in a given column.

SELECT SUM(price)
FROM items_ordered

items_ordered

customerid	order_date	item	quantity	price
10330	30-Jun-1999	Pogo stick	1	28.00
10101	30-Jun-1999	Raft	1	58.00
10298	01-Jul-1999	Skateboard	1	33.00
10101	01-Jul-1999	Life Vest	4	125.00
10299	06-Jul-1999	Parachute	1	1250.00
10339	27-Jul-1999	Umbrella	1	4.50
10449	13-Aug-1999	Unicycle	1	180.79
10439	14-Aug-1999	Ski Poles	2	25.50
10101	18-Aug-1999	Rain Coat	1	18.30

Syntax

SELECT SUM(<column_name>)

FROM <table_name>

WHERE<condition>

SELECT MIN Statement

Sample

The **MIN()** function returns the smallest value in a given column.

SELECT MIN(price)
FROM items_ordered

items_ordered

customerid	order_date	item	quantity	price
10330	30-Jun-1999	Pogo stick	1	28.00
10101	30-Jun-1999	Raft	1	58.00
10298	01-Jul-1999	Skateboard	1	33.00
10101	01-Jul-1999	Life Vest	4	125.00
10299	06-Jul-1999	Parachute	1	1250.00
10339	27-Jul-1999	Umbrella	1	4.50
10449	13-Aug-1999	Unicycle	1	180.79
10439	14-Aug-1999	Ski Poles	2	25.50
10101	18-Aug-1999	Rain Coat	1	18.30

Syntax

SELECT MIN(<column_name>)

FROM <table_name>

WHERE<condition>

SELECT MAX Statement

Sample

The **MAX()** function returns the largest value in a given column.

SELECT MAX(price)
FROM items_ordered

items_ordered

customerid	order_date	item	quantity	price
10330	30-Jun-1999	Pogo stick	1	28.00
10101	30-Jun-1999	Raft	1	58.00
10298	01-Jul-1999	Skateboard	1	33.00
10101	01-Jul-1999	Life Vest	4	125.00
10299	06-Jul-1999	Parachute	1	1250.00
10339	27-Jul-1999	Umbrella	1	4.50
10449	13-Aug-1999	Unicycle	1	180.79
10439	14-Aug-1999	Ski Poles	2	25.50
10101	18-Aug-1999	Rain Coat	1	18.30

Syntax

SELECT MAX(<column_name>)

FROM <table_name>

WHERE<condition>

SELECT AVG Statement

Sample

The **AVG()** function returns the average value of a given column.

SELECT AVG(price)
FROM items_ordered

items_ordered

customerid	order_date	item	quantity	price
10330	30-Jun-1999	Pogo stick	1	28.00
10101	30-Jun-1999	Raft	1	58.00
10298	01-Jul-1999	Skateboard	1	33.00
10101	01-Jul-1999	Life Vest	4	125.00
10299	06-Jul-1999	Parachute	1	1250.00
10339	27-Jul-1999	Umbrella	1	4.50
10449	13-Aug-1999	Unicycle	1	180.79
10439	14-Aug-1999	Ski Poles	2	25.50
10101	18-Aug-1999	Rain Coat	1	18.30

Syntax

SELECT AVG(<column_name>)

FROM <table_name>

WHERE<condition>

SELECT DISTINCT Statement

The **SELECT DISTINCT** statement is used to return only distinct (different) values.

Inside a table, a column often contains many duplicate values; and sometimes you only want to list the different (distinct) values.

Syntax

SELECT DISTINCT <attributes>

FROM <table_name>

SELECT DISTINCT Statement

```
SELECT DISTINCT Country
FROM Customers;
```

What will be the output of the query?

CustomerID	CustomerName	Address	City	Country
1	Alfreds Futterkiste	Obere Str. 57	Berlin	Germany
2	Ana Trujillo Emparedados y helados	Avda. de la Constitución 2222	México D.F.	Mexico
3	Antonio Moreno Taquería	Mataderos 2312	México D.F.	Mexico
4	Around the Horn	120 Hanover Sq.	London	UK
5	Berglunds snabbköp	Berguvsvägen 8	Luleå	Sweden

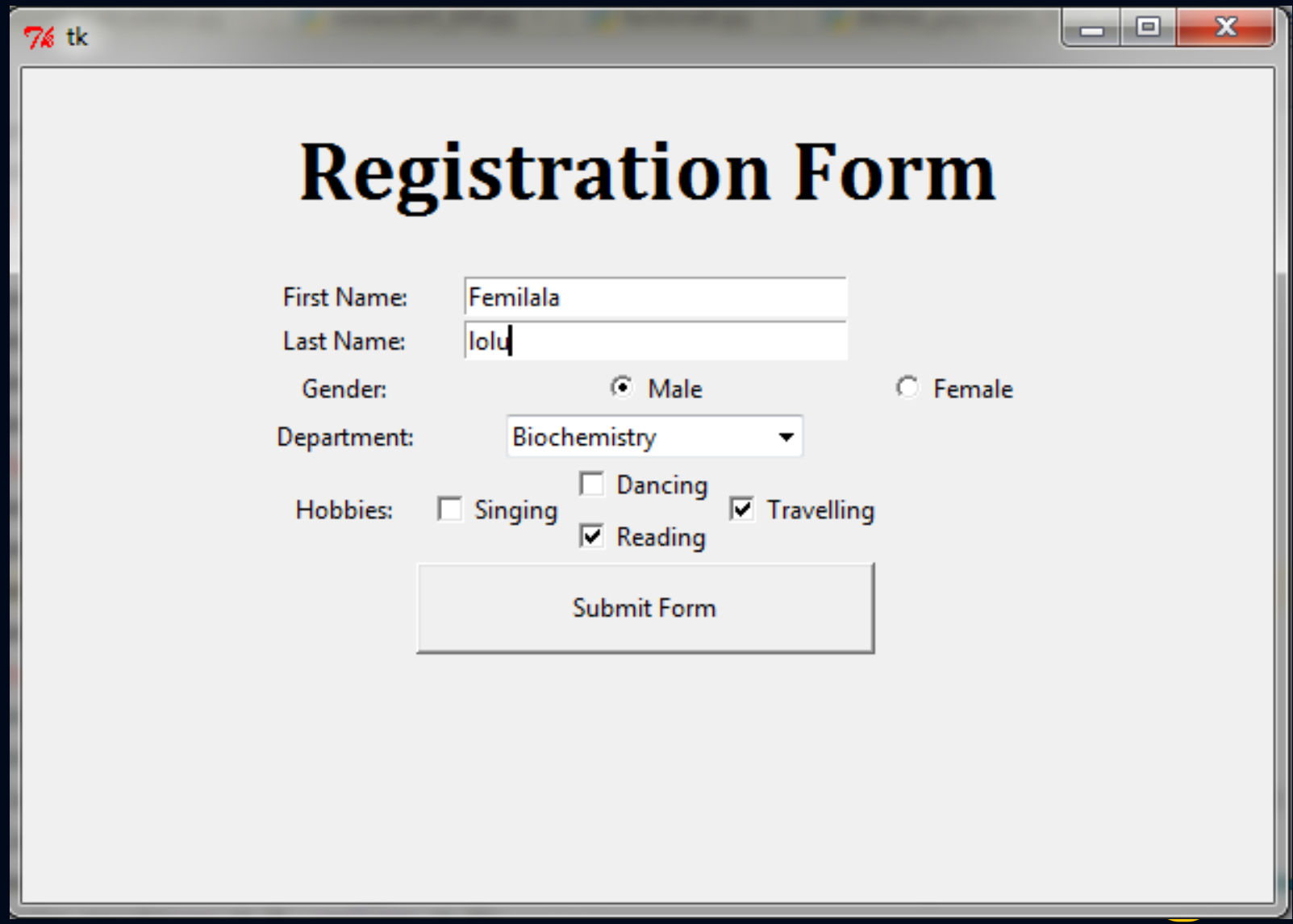
Basic CRUD Application

A **CRUD** application is one that uses forms to get data into and out of a database.

Basic CRUD Application

Output

We shall develop a registration form that accepts user data and save in a database.



Basic CRUD Application

```
from tkinter import ttk
from tkinter import *
#create main window
window = Tk()
window.geometry("600x400")
lbl_header = Label(text = 'Registration Form', font = 'Cambria 30 bold', pady = 20)
lbl_header.pack(fill = X)
#create 2 frames
frm_east = Frame(window, bd = 5, height = 400, width = 250)
frm_east.pack(side = TOP)
#create widgets for first_name
fNameVar,lNameVar, genderVar = StringVar(), StringVar(), StringVar()
lbl_fName = Label(frm_east, text = 'First Name: ')
lbl_fName.grid(row = 0, column = 0)
ent_fName = Entry(frm_east, textvariable = fNameVar, width = 30)
ent_fName.grid(row = 0, column = 1)
#widgets for last_name
lbl_lName = Label(frm_east, text = 'Last Name: ')
lbl_lName.grid(row = 1, column = 0)
ent_lName = Entry(frm_east, textvariable = lNameVar, width = 30)
ent_lName.grid(row = 1, column = 1)
```

creates widgets and bind to main window

Basic CRUD Application

Block of code below creates widgets and bind to main window

```
#widgets for gender
lbl_gender = Label(frm_east, text = 'Gender: ')
lbl_gender.grid(row = 2, column = 0)
rad_male = Radiobutton(frm_east, text = 'Male', variable=genderVar, value = 'Male')
rad_male.grid(row = 2, column = 1)
rad_female = Radiobutton(frm_east, text = 'Female', variable=genderVar, value = 'Female')
rad_female.grid(row = 2, column = 2)
#widgets for department
deptVar = StringVar()
lbl_dept = Label(frm_east, text = 'Department: ')
lbl_dept.grid(row = 3, column = 0)
dept = ('CIS','MIS','Physics','Maths')
cmb_dept = ttk.Combobox(frm_east, values=dept, textvariable = deptVar)
cmb_dept.grid(row = 3, column = 1)
```

Basic CRUD Application

```
#widgets for hobbies
checkvar1,checkvar2,checkvar3,checkvar4 = StringVar(),StringVar(),StringVar(),StringVar()
lbl_hobby = Label(frm_east, text = 'Hobbies: ')
lbl_hobby.grid(row = 4, column = 0)
frm_hobby = Frame(frm_east)
frm_hobby.grid(row = 4, column = 1)
singing = Checkbutton(frm_hobby, text="Singing", variable = checkvar1, onvalue='Singing', offvalue= '')
travelling = Checkbutton(frm_hobby, text="Travelling", variable = checkvar2, onvalue='Travelling', offvalue= '')
reading = Checkbutton(frm_hobby, text="Reading", variable = checkvar3, onvalue='Reading', offvalue= '')
dancing = Checkbutton(frm_hobby, text="Dancing", variable = checkvar4, onvalue='Dancing', offvalue= '')
singing.pack(side=LEFT)
travelling.pack(side=RIGHT)
reading.pack(side=BOTTOM)
dancing.pack(side=BOTTOM)
#widget submit button
btn_submit = Button(frm_east, text = 'Submit Form', command = create_table, width = 30, pady = 10)
btn_submit.grid(row = 5, columnspan = 5)

window.mainloop()
```

creates widgets and bind to main window

Basic CRUD Application


```
from tkinter import messagebox
import sqlite3
conn = sqlite3.connect('sample.db')
c = conn.cursor()
def create_table():
    c.execute("CREATE TABLE IF NOT EXISTS register (id INTEGER PRIMARY KEY AUTOINCREMENT, fName TEXT, "
              "lName TEXT, gender TEXT, dept TEXT, hobbies TEXT)")
    conn.commit()
    add_data()

def add_data():
    #accept user input values and into variables
    lst_hobbies = [checkvar1.get(),checkvar2.get(),checkvar3.get(),checkvar4.get()] #place all hobbies in a list
    hobbies = [item for item in lst_hobbies if item]  #Remove unchecked hobbies from the list of hobbies
    hobbies = ' '.join(map(str, hobbies))  #covert a list object to a string object before saving to DB

    #Validate your input data here to prevent unwanted data input

    c.execute("INSERT INTO register (fName,lName,gender,dept,hobbies) VALUES(?,?,?,?,?)",
              (fNameVar.get(),lNameVar.get(),genderVar.get(),deptVar.get(),hobbies))
    conn.commit()
    messagebox.showinfo('Registration Form', 'Record Saved Successfully!')
```

Next Lecture ...

Python  Programming

Day 17: Introduction to Web Programming