Code :

Let’s begin writing the code in steps .

Initially As we are using python for creating the user interface , So we first have to connect python to our mysql for which the code is as follows :

**import mysql.connector**

**from datetime import datetime, timedelta**

**from random import choice**

**mydb = mysql.connector.connect(host = "localhost", user = "root", passwd = "ADARSH1")**

**cur = mydb.cursor()**

Now the next step is to create a database for the storing and managing data from the normalized tables that we have got from above .

Then we go on to create all the above fore mentioned tables .

**def create\_Database():**

**qry = "CREATE DATABASE IF NOT EXISTS dbms\_project"**

**cur.execute(qry)**

**qry = "USE dbms\_project"**

**cur.execute(qry)**

**qry = "CREATE TABLE IF NOT EXISTS PATIENT\_INFO ( " \**

**"Pat\_ID INT PRIMARY KEY NOT NULL auto\_increment, " \**

**"Name VARCHAR(255), " \**

**"Age INT, " \**

**"Arrived\_From VARCHAR(255), " \**

**"Leaving\_To VARCHAR(255)) "**

**cur.execute(qry)**

**qry = "CREATE TABLE IF NOT EXISTS ADM\_PERIOD ( " \**

**"Pat\_ID INT REFERENCES PATIENT\_INFO(Pat\_ID) ON DELETE CASCADE, " \**

**"Adm\_Date DATE, " \**

**"Dis\_Date DATE) "**

**cur.execute(qry)**

**qry = "CREATE TABLE IF NOT EXISTS PATIENT\_ADDRESS ( " \**

**"Pat\_ID INT REFERENCES PATIENT\_INFO(Pat\_ID) ON DELETE CASCADE, " \**

**"House\_Num VARCHAR(255), " \**

**"Street VARCHAR(255), " \**

**"Area VARCHAR(255), " \**

**"City VARCHAR(255), " \**

**"Pincode INT, " \**

**"State VARCHAR(255), " \**

**"Country VARCHAR(255)) "**

**cur.execute(qry)**

**qry = "CREATE TABLE IF NOT EXISTS PATIENT\_CONTACT ( " \**

**"Pat\_ID INT REFERENCES PATIENT\_INFO(Pat\_ID) ON DELETE CASCADE, " \**

**"Phone\_Num VARCHAR(255)) "**

**cur.execute(qry)**

**qry = "CREATE TABLE IF NOT EXISTS ROOM\_INFO ( " \**

**"Pat\_ID INT REFERENCES PATIENT\_INFO(Pat\_ID) ON DELETE CASCADE, " \**

**"Hostel\_Num INT, " \**

**"Floor\_Num INT, " \**

**"Room\_Num INT) "**

**cur.execute(qry)**

**mydb.commit()**

Now every time we have to insert a record in a table , we are going to call insert\_Record() and it will insert data into all the table using mysql commands .

**def insert\_Record():**

**# Insert Patient Info , Phone Numbers , Patient Address , Room Info**

**qry = "INSERT INTO PATIENT\_INFO (NAME, Age, Arrived\_From, Leaving\_To) " \**

**"VALUES (%s, %s, %s, %s) "**

**name = input("Enter Patient Name : ")**

**age = int(input("Enter Age : "))**

**arr\_frm = input("From where did you come? ")**

**lev\_to = input("Where will you be heading to? ")**

**cur.execute(qry, (name, age, arr\_frm, lev\_to))**

**mydb.commit()**

**qry = "INSERT INTO PATIENT\_CONTACT (Pat\_ID, Phone\_Num) " \**

**" VALUES (%s, %s) "**

**phone\_num = [int(x) for x in input("Enter Phone Numbers separated by space : ").split()]**

**data = []**

**getqry = "SELECT LAST\_INSERT\_ID() "**

**cur.execute(getqry)**

**pat\_id = 0**

**for x in cur:**

**for id in x:**

**pat\_id = id**

**for num in phone\_num:**

**data.append((pat\_id, num))**

**cur.executemany(qry, data)**

**qry = "INSERT INTO PATIENT\_ADDRESS (Pat\_ID, House\_Num, Street, Area, City, Pincode, State, Country) " \**

**"VALUES (%s, %s, %s, %s, %s, %s, %s, %s) "**

**house\_num = input("Enter House Number : ")**

**street = input("Enter Street Name : ")**

**area = input("Enter Area Name : ")**

**city = input("Enter City Name : ")**

**pincode = int(input("Enter Area's Pincode : "))**

**state = input("Enter State : ")**

**country = input("Enter Country : ")**

**cur.execute(qry, (pat\_id, house\_num, street, area, city, pincode, state, country))**

**qry = "INSERT INTO ROOM\_INFO (Pat\_ID, Hostel\_Num, Floor\_Num, Room\_Num) " \**

**"VALUES (%s, %s, %s, %s) "**

**room\_num = 0;**

**if age >= 60:**

**room\_num = choice(room[0])**

**room[0].remove(room\_num)**

**elif age >= 40:**

**room\_num = choice(room[1])**

**room[1].remove(room\_num)**

**else:**

**room\_num = choice(room[2])**

**room[2].remove(room\_num)**

**floor\_num = (room\_num // 100) % 10**

**hostel\_num = room\_num // 1000**

**room\_num %= 100**

**cur.execute(qry, (pat\_id, hostel\_num, floor\_num, room\_num))**

**mydb.commit()**

**# Display alloted room**

**print("Assigned Hostel " + str(hostel\_num) + " Floor " + str(floor\_num) + " Room " + str(room\_num) + " to Patient " + str(name) + " (ID : " + str(pat\_id) + ")")**

**print("")**

Now we also want to extract information about any person quarantined in the hostel at any moment . For that purpose what we are trying to do here is to find records of the person using his allotted patient Id and returning the information of the person in a map/dictionary mp .

**def getData(id):**

**mp = {}**

**qry = "SELECT \* FROM PATIENT\_INFO WHERE Pat\_ID = %s "**

**cur.execute(qry, (id, ))**

**for x in cur:**

**mp["pat\_id"] = x[0]**

**mp["name"] = x[1]**

**mp["age"] = x[2]**

**mp["com\_frm"] = x[3]**

**mp["lev\_to"] = x[4]**

**qry = "SELECT \* FROM ADM\_PERIOD WHERE Pat\_ID = %s "**

**cur.execute(qry, (id, ))**

**for x in cur:**

**mp["adm\_date"] = x[1]**

**mp["dis\_date"] = x[2]**

**qry = "SELECT Phone\_Num FROM PATIENT\_CONTACT WHERE Pat\_ID = %s "**

**cur.execute(qry, (id, ))**

**mp["phn"] = []**

**for x in cur:**

**for num in x:**

**mp["phn"].append(num)**

**qry = "SELECT \* FROM PATIENT\_ADDRESS WHERE Pat\_ID = %s "**

**cur.execute(qry, (id, ))**

**for x in cur:**

**mp["house\_num"] = x[1]**

**mp["street"] = x[2]**

**mp["area"] = x[3]**

**mp["city"] = x[4]**

**mp["pincode"] = x[5]**

**mp["state"] = x[6]**

**mp["country"] = x[7]**

**qry = "SELECT \* FROM ROOM\_INFO WHERE Pat\_id = %s "**

**cur.execute(qry, (id, ))**

**for x in cur:**

**mp["hostel\_num"] = x[1]**

**mp["floor\_num"] = x[2]**

**mp["room\_num"] = x[3]**

**return mp**

Now in order to display the data for a given patient using the map/dictionary mp :

**def showData(mp):**

**# This function prints the data using the dictionary**

**print("")**

**print("Patient Id : " + str(mp["pat\_id"]))**

**print("Name : " + str(mp["name"]))**

**print("Age : " + str(mp["age"]))**

**print("Coming From : " + str(mp["com\_frm"]))**

**print("Leaving To : " + str(mp["lev\_to"]))**

**print("Admission Date : " + str(mp["adm\_date"]))**

**print("Discharge Date : " + str(mp["dis\_date"]))**

**print("Address : " + str(mp["house\_num"]) + " " + str(mp["street"]) + " " + str(mp["area"]) + ", " + str(mp["city"]) + ", " + str(mp["state"]) + ", " + str(mp["country"]))**

**print("PinCode : " + str(mp["pincode"]))**

**print("Phone Numbers :", \*mp["phn"])**

**print("Hostel : " + str(mp["hostel\_num"]) + ", Floor : " + str(mp["floor\_num"]) + ", Room : " + str(mp["room\_num"]))**

**print("")**

**print("---------------------------------------------------------------------")**

Now to view records of a person initially , we have to either know allotted person’s ID or we need to know the name .

**def view\_Records():**

**qry = "SELECT Pat\_ID from PATIENT\_INFO "**

**cur.execute(qry)**

**pat\_id = []**

**for x in cur:**

**for id in x:**

**pat\_id.append(id)**

**pat\_id.sort()**

**if len(pat\_id) == 0:**

**print("No records found")**

**else:**

**data = []**

**for id in pat\_id:**

**mp = getData(id)**

**data.append(mp)**

**for mp in data:**

**showData(mp)**

**print("")**

**def search\_Record():**

**# This function is to print the data of a particular patient, either**

**# by id or by name. It then uses getData and showData functions to**

**# get and print the data**

**c = int(input("Enter \n1 to search using Patient ID\n2 to Search using Name : "))**

**pat\_id = []**

**if(c == 1):**

**x = int(input("Enter Patient ID : "))**

**getqry = "SELECT Pat\_ID FROM PATIENT\_INFO WHERE Pat\_ID = %s "**

**cur.execute(getqry, (x, ))**

**for x in cur:**

**pat\_id.append(int(x[0]))**

**else:**

**name = input("Enter Patient Name : ")**

**getqry = "SELECT Pat\_ID FROM PATIENT\_INFO WHERE Name = %s "**

**cur.execute(getqry, (name, ))**

**for x in cur:**

**for id in x:**

**pat\_id.append(int(id))**

**data = []**

**for id in pat\_id:**

**mp = getData(id)**

**data.append(mp)**

**if len(data) == 0:**

**print("No records found")**

**else:**

**for mp in data:**

**showData(mp)**

**print("")**

Now we might need to delete some records which might be redundant after the person leaves the premise , for that purpose delete\_Record() is being called , which deletes the record from all the tables .

**def delete\_Record():**

**# Delete record from all 5 tables where Pat\_ID = id**

**# Add the room number of discharged patient back to vacant rooms**

**pat\_id = input("Enter the Patient ID : ")**

**qry = "SELECT Hostel\_num, Floor\_Num, Room\_Num from ROOM\_INFO "**

**cur.execute(qry)**

**room\_num, floor\_num, hostel\_num = 0, 0, 0**

**for x in cur:**

**room\_num = x[0]**

**floor\_num = x[1]**

**room\_num = x[2]**

**if room\_num == 0:**

**return**

**room\_num = int(hostel\_num) \* 1000 + int(floor\_num) \* 100 + int(room\_num)**

**room[floor\_num].append(room\_num)**

**qry = "DELETE FROM PATIENT\_INFO WHERE Pat\_ID = %s "**

**cur.execute(qry, (pat\_id, ))**

**qry = "DELETE FROM PATIENT\_ADDRESS WHERE Pat\_ID = %s "**

**cur.execute(qry, (pat\_id, ))**

**qry = "DELETE FROM PATIENT\_CONTACT WHERE Pat\_ID = %s "**

**cur.execute(qry, (pat\_id, ))**

**qry = "DELETE FROM ROOM\_INFO WHERE Pat\_ID = %s "**

**cur.execute(qry, (pat\_id, ))**

**qry = "DELETE FROM ADM\_PERIOD WHERE Pat\_id = %s "**

**cur.execute(qry, (pat\_id, ))**

**mydb.commit()**

**print("Record deleted successfully")**

**print("")**

This function calls all the triggers to start working .

**def use\_triggers():**

This function capitalizes the first letter of the name .

**qry = "DROP FUNCTION IF EXISTS camelcase "**

**cur.execute(qry)**

This function takes the string and scans it from left to right

As it encounters a puncutation, it capitalizes the next character

**qry = "CREATE FUNCTION `camelcase`( str VARCHAR(255) ) RETURNS VARCHAR(255) CHARSET utf8 DETERMINISTIC BEGIN DECLARE c CHAR(1); DECLARE s VARCHAR(255); DECLARE i INT DEFAULT 1; DECLARE bool INT DEFAULT 1; DECLARE punct CHAR(17) DEFAULT ' ()[]{},.-\_!@;:?/'; SET s = LCASE( str ); WHILE i < LENGTH( str ) DO BEGIN SET c = SUBSTRING( s, i, 1 ); IF LOCATE( c, punct ) > 0 THEN SET bool = 1; ELSEIF bool=1 THEN BEGIN IF c >= 'a' AND c <= 'z' THEN BEGIN SET s = CONCAT(LEFT(s,i-1),UCASE(c),SUBSTRING(s,i+1)); SET bool = 0; END; ELSEIF c >= '0' AND c <= '9' THEN SET bool = 0; END IF; END; END IF; SET i = i+1; END; END WHILE; RETURN s; END"**

**cur.execute(qry)**

**mydb.commit()**

Creating a trigger to convert name to camel case when inserting a record

**qry = "DROP TRIGGER IF EXISTS cap\_on\_insert "**

**cur.execute(qry)**

**qry = "CREATE TRIGGER cap\_on\_insert BEFORE INSERT ON PATIENT\_INFO FOR EACH ROW BEGIN SET NEW.Name = camelcase(NEW.Name); END; "**

**cur.execute(qry)**

Creating a trigger to convert country, state, city, area and street

to camel case when inserting a record

**qry = "DROP TRIGGER IF EXISTS cap\_address "**

**cur.execute(qry)**

**qry = "CREATE TRIGGER cap\_address BEFORE INSERT ON PATIENT\_ADDRESS FOR EACH ROW BEGIN SET NEW.Country = camelcase(NEW.Country), NEW.State = camelcase(NEW.State), NEW.city = camelcase(NEW.city), NEW.area = camelcase(NEW.area), NEW.street = camelcase(NEW.street); END; "**

**cur.execute(qry)**

Inserting admission date and discharge date to admission period table

as a new record is inserted into patient details table

**qry = "DROP TRIGGER IF EXISTS ins\_adm\_period "**

**cur.execute(qry)**

**qry = "CREATE TRIGGER ins\_adm\_period AFTER INSERT ON PATIENT\_INFO FOR EACH ROW BEGIN INSERT INTO ADM\_PERIOD(Pat\_Id, Adm\_Date, Dis\_Date) VALUES (NEW.Pat\_ID, CURDATE(), DATE\_ADD(CURDATE(), INTERVAL 14 DAy)); END; "**

**cur.execute(qry)**

**mydb.commit()**

Allocation for rooms is being done by 2D array room with appending rooms which can be occupied .

**room = [[] for \_ in range(3)]**

**def roomAllocation():**

**for r in range(1000, 1100):**

**room[0].append(r)**

**for r in range(2000, 2100):**

**room[0].append(r)**

**for r in range(1100, 1200):**

**room[1].append(r)**

**for r in range(2100, 2200):**

**room[1].append(r)**

**for r in range(1200, 1250):**

**room[2].append(r)**

**for r in range(2200, 2250):**

**room[2].append(r)**

Calling all the above functions for creating database , allocating rooms and making triggers start working .

**roomAllocation()**

**create\_Database()**

**use\_triggers()**

This piece of code is for removal of rooms which are being still occupied by the people residing in the hostel rooms .

**qry = "SELECT Hostel\_Num, Floor\_Num, Room\_Num from ROOM\_INFO "**

**cur.execute(qry)**

**for x in cur:**

**room[x[1]].remove(int(x[0]) \* 1000 + int(x[1]) \* 100 + int(x[2]))**

Now this piece of code gives a user interface with options to pick when the code runs .

**while True:**

**c = int(input("Enter \n1 to Insert a Patient's Details\n2 to View all Records\n3 to Search a Patient \n4 to Delete a Patient's Details\n5 to Exit : "))**

**if c == 1:**

**insert\_Record()**

**elif c == 2:**

**view\_Records()**

**elif c == 3:**

**search\_Record()**

**elif c == 4:**

**delete\_Record()**

**else:**

**break**

Closing connection to end the program .

**cur.close()**

**mydb.close()**