COMPUTER SCIENCE (083) - PROJECT REPORT

WEDDING PLANNING

with Python and Mysql

2021 AISSCE EXAMINATION

TABLE OF CONTENTS

ABSTRACT

Wedding planning takes a lot of work. There is a guest list to be confirmed, a buffet to be ordered, a venue to be secured, photographers to be hired, and entertainment to be chartered - all within a fixed budget. Thus, in our project, we use a database to keep track of the number of attendees and calculate the total expenditure accordingly with an easy-to-use program that offers the user a variety of vendors to choose from.

We use MySQL to create several tables to do the same – store details of the guests, the vendors, and the expenses. A code is then written up using Python to insert data into tables based on the user's input, delete redundant information, and perform basic calculations after retrieving data. In the process of doing so, we will be integrating Python with MySQL; the data is stored on the MySQL server while Python acts as an interface through which we will be accessing the data.

Let the wedding planning begin!

PROJECT AIM

The aim of this project is to make the wedding planning process as efficient as possible. Ideally, all the user has to do is enter their budget and guest list. This software takes care of the rest.

With a function to calculate the number of confirmed attendees (based on RSVPs), and several other functions to pick the best outsourcers (based on user preferences, this software is essentially an optimization program that works to save as much time and effort as possible. In summary, our objective is to:

- enhance efficiency by automating the process and eliminating the need to approach different vendors and services independently - serves as a one-stop wedding planning service

- optimise the wedding planning process by saving time and effort as possible with a pick-and-choose program

- create an easily replicable and accessible (user-friendly) program that can be used across most platforms with no prior training

LIMITATIONS

There are some limitations to this program:

- It does not take into account other expenses such as hiring florists, paying for the wedding dress and tuxedo (and fitting sessions), and putting together the wedding favours (thank-you gift bags)

- Another assumption that has been made is that invitations have been sent out to everyone on the guest list despite the lack of code to automate that part of the process. We start the process assuming the guests have responded to the invitations

- The database has four tables to keep tabs of all the details of the vendors. However, there are not many records that have been inserted. Thus, there are only ten options from each vendor that the user can choose from

SYSTEM

SOFTWARE USED:

Operating system: Microsoft Windows 10 Pro
Version 20H2, build: 19042.746

Front end development environment: PyCharm
Community Edition 2020.3.1

Database software: MySQL Shell 8.0

Server host: smartASP.net

Documentation: Canva

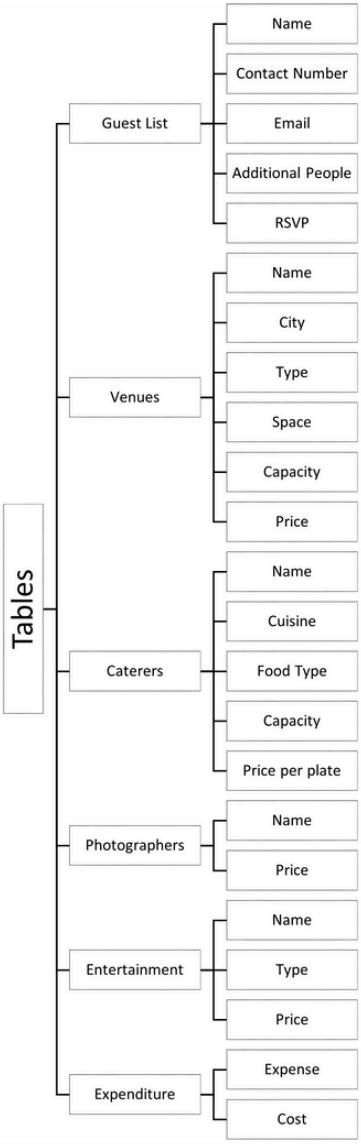
HARDWARE USED:

Processor: Intel(R) Pentium(R) CPU G640 @ 2.80GHz

Other computer peripherals: Keyboard, mouse

System type: 64-bit operating system

Installed RAM: 8.00 GB



TASK 1
Create the guest list by getting input (name, email, etc.) from the user

TASK 2
Calculate the total number of guests based on the RSVPs and update the table

TASK 3
Pick the best wedding vendors based on user preferences

TASK 4
Calculate the total expenditure and state if user is within their assigned budget

SOURCE CODE

TASK 1

Assuming the wedding invitations have been sent out, and that the user has received the RSVPs and the details of the number of additional people that each guest is bringing, the table **Guest List** is created. The function that does this is defined as follows:

```
def guestlist():
    # getting input details from the user
    numofinvites = int(input("How many guests are you inviting?"))
    for i in range(0, numofinvites):
        guestname = input("Enter the name of the guest")
        guestemail = input("Enter the guest's email")
        guestnum = int(input("Enter the guest's contact number"))
        guestRSVP = input("Is the guest attending? Enter 'yes' or 'no'")

        # validation check for those who aren't attending
        if guestRSVP == "no":
            addpeople = 0
        else:
            addpeople = int(input("Enter the number of additional people this guest is bringing"))

        # adding the records into the table
        guestvalues = "insert into guestlist (Name, Number, RSVP, Additional_People,Email) \
                  values ('{}','{}','{}','{}','{}')".format(guestname, guestnum, guestRSVP, addpeople, guestemail)
        print(guestvalues)
        cursor.execute(guestvalues)
        mycon.commit()

    # printing the records for convenience
    cursor.execute("select * from guestlist")
    data = cursor.fetchall()
    print("Guest list successfully updated!")
    details = input("Do you want to see the guest list? Enter 'y' or 'n'")
    if details == "y":
        for row in data:
            print(row)
```

The table Guest List is created as shown:

Field	Type	Null	Key	Default	Extra
Name	varchar(50)	NO		NULL	
Number	int	NO	PRI	NULL	
Email	varchar(50)	YES		NULL	
RSVP	varchar(3)	YES		Yes	
Food_Preference	varchar(20)	YES		NULL	
Additional_People	int	YES		0	

6 rows in set (0.2762 sec)

Name	Number	Email	RSVP	Food_Preference	Additional_People
Sheila	566107723	sheilazazz@gmail.com	yes	Non-vegetarian	2
Tara	657887653	taramusich@outlook.com	yes	Non-vegetarian	1
Troye	677512210	troyesivan@outlook.com	no	Non-vegetarian	0
Jess	678994516	jessstamos@gmail.com	no	Non-vegetarian	0
Rose	700981233	rosemeghan@gmail.com	yes	Vegetarian	2
John	766564532	johnpierre@gmail.com	no	Non-vegetarian	0
Siddarth	784107533	siddarthsundar@gmail.com	yes	Non-vegetarian	0
Kyra	786767564	kyrapatel@gmail.com	yes	Non-vegetarian	0
David	787865465	davidjoe@gmail.com	no	Non-vegetarian	0
Carlos	788091132	carlosrodriguez@gmail.com	yes	Vegetarian	2
Mike	788091232	mikeross@gmail.com	yes	Non-vegetarian	2
Grey	788675432	gerychance@outlook.com	no	Non-vegetarian	0
Raj	788833510	rajmagesh@gmail.com	yes	Vegetarian	4
Ari	788865102	arilauv@gmail.com	no	Vegetarian	0
Meena	788866710	smeena@gmail.com	yes	Non-vegetarian	4
Kara	788893220	karadanvers@gmail.com	yes	Non-vegetarian	3
Trish	788909122	trishdelarosa@outlook.com	yes	Non-vegetarian	3
Charlie	788987432	charlier@outlook.com	yes	Vegetarian	4
Felicity	800765321	felicitysmoaks@outlook.com	no	Non-vegetarian	0
Alex	877608221	alexjohn@gmail.com	yes	Vegetarian	2
Julia	877654432	juliamichaels@outlook.com	no	Non-vegetarian	0
Gowri	877786544	gowrirv@gmail.com	yes	Non-vegetarian	1
Camila	878897656	camilamendes@gmail.com	yes	Vegetarian	2
Harvey	889011524	harveyspecter@outlook.com	yes	Non-vegetarian	2
Maya	893900624	mayavarsh@gmail.com	yes	Non-vegetarian	1
Ross	899893321	rosslynch@gmail.com	yes	Vegetarian	3
Gman	899897610	kgman@microsoft.com	yes	Non-vegetarian	3
Alessia	900766321	alessiacara@gmail.com	yes	Vegetarian	2
Calum	900787811	calumworthy@gmail.com	yes	Vegetarian	2
Oliver	900854461	oliverqueen@gmail.com	no	Non-vegetarian	0
Yusuf	900874332	yusufahmed@gmail.com	no	Non-vegetarian	0
Chloe	900876671	chloedecker@gmail.com	yes	Vegetarian	2
Laura	900876761	lauramarano@outlook.com	yes	Non-vegetarian	3
Nina	907765221	niniroberts@gmail.com	yes	Non-vegetarian	3
Meghan	908871123	meghanrienks@outlook.com	yes	Non-vegetarian	3
Ricky	908876123	rickybowen@hotmail.com	yes	Vegetarian	2
Alice	908876523	alicemain@microsoft.com	yes	Vegetarian	3
Jay	984107334	jayquellin@outlook.com	yes	Non-vegetarian	2
Rhea	987787875	rheanair@gmail.com	no	Vegetarian	0
Lily	987878654	lilymarston@gmail.com	yes	Non-vegetarian	1

TASK 2

Now that we have a record of all the guests and their details, we have to calculate the total number of people that are confirmed to attend. This involves the adding the number of invitees on the guest list who responded with a 'yes' when asked to RSVP, and the number of additional people that each guest is bringing (if any). The function that does this is defined as follows:

```
# function2
def totalguests():
    # counting the total number of people who responded with "yes"
    cursor.execute("select count(*) from guestlist where RSVP = 'yes'")
    gyes = cursor.fetchall()
    guestyesnum = int(''.join(map(str, gyes[0])))
    print("The number of invitees who responded with 'yes' is", guestyesnum)

    # counting the total number of additional guests
    cursor.execute("select sum(additional_people) from guestlist")
    addppl = cursor.fetchall()
    addpplnum = int(''.join(map(str, addppl[0])))
    print("The number of additional guests is", addpplnum)

    # counting the total number of guests
    total_guests = guestyesnum + addpplnum
    print("The total number of attendees is", total_guests)

    # updating the table - deleting the records of those guests who aren't attending
    cursor.execute("delete from guestlist where RSVP = 'no'")
    cursor.execute("select * from guestlist")
    confirmedguestlist = cursor.fetchall()
    details = input("Do you want to see the updated guest list? Enter 'y' or 'n'")
    if details == 'y':
        for row in confirmedguestlist:
            print(row)

    return guestyesnum, total_guests
```

Once the total number of guests has been calculated, the table is updated - the records of those who aren't attending are delete as shown:

Name	Number	Email	RSVP	Food_Preference	Additional_People
Sheila	566107723	sheilazazz@gmail.com	yes	Non-vegetarian	2
Tara	657887653	taramusich@outlook.com	yes	Non-vegetarian	1
Rose	700981233	rosemeghan@gmail.com	yes	Vegetarian	2
Siddarth	784107533	siddarthsundar@gmail.com	yes	Non-vegetarian	0
Kyra	786767564	kyrapatel@gmail.com	yes	Non-vegetarian	0
Carlos	788091132	carlosrodriguez@gmail.com	yes	Vegetarian	2
Mike	788091232	mikeross@gmail.com	yes	Non-vegetarian	2
Raj	788833510	rajmagesh@gmail.com	yes	Vegetarian	4
Meena	788866710	smeena@gmail.com	yes	Non-vegetarian	4
Kara	788893220	karadanvers@gmail.com	yes	Non-vegetarian	3
Trish	788909122	trishdelarosa@outlook.com	yes	Non-vegetarian	3
Charlie	788987432	charliert@outlook.com	yes	Vegetarian	4
Alex	877608221	alexjohn@gmail.com	yes	Vegetarian	2
Gowri	877786544	gowrirv@gmail.com	yes	Non-vegetarian	1
Camila	878897656	camilamendes@gmail.com	yes	Vegetarian	2
Harvey	889011524	harveyspecter@outlook.com	yes	Non-vegetarian	2
Maya	893900624	mayavarsh@gmail.com	yes	Non-vegetarian	1
Ross	899893321	rosslynch@gmail.com	yes	Vegetarian	3
Gman	899897610	kgman@microsoft.com	yes	Non-vegetarian	3
Alessia	900766321	alessiacara@gmail.com	yes	Vegetarian	2
Calum	900787811	calumworthy@gmail.com	yes	Vegetarian	2
Chloe	900876671	chloedecker@gmail.com	yes	Vegetarian	2
Laura	900876761	lauramarano@outlook.com	yes	Non-vegetarian	3
Nina	907765221	niniroberts@gmail.com	yes	Non-vegetarian	3
Meghan	908871123	meghanrienks@outlook.com	yes	Non-vegetarian	3
Ricky	908876123	rickybowen@hotmail.com	yes	Vegetarian	2
Alice	908876523	alicemain@microsoft.com	yes	Vegetarian	3
Maya	917008251	mayajay@gmail.com	yes	NULL	2
Maya	917600475	mayavarsh@gmail.com	yes	NULL	3
Lily	987878654	lilymarston@gmail.com	yes	Non-vegetarian	1
Rohith	988787651	rohithkanna@gmail.com	yes	Vegetarian	1
Jake	988787654	jakeryan@gmail.com	yes	Non-vegetarian	3
Aurelia	988829102	aureliawest@gmail.com	yes	Vegetarian	2
Shawn	989876765	shawnmendes@gmail.com	yes	Non-vegetarian	1
Ally	990076122	allydawson@gmail.com	yes	Vegetarian	3
Austin	990871162	austinmoon@gmail.com	yes	Vegetarian	2
Rachel	990878162	racheladams@gmail.com	yes	Vegetarian	2

Thus, in tasks 1 and 2, we have successfully **updated a table** (that is, inserted values into it), and **deleted records from a table** (eliminated redundant data) by integrating Python with MySQL.

TASK 3

In task 3, we define functions to pick the best venue, caterer, photographer, and entertainer respectively. To do this, we first created tables that stored data of different companies for each expense. For instance, we created the table **Caterers** and updated it to include the details of 10 catering crews that the user could choose from. Similarly, details of 10 venues, studios, and entertainers were updated into their respective tables . Thus, we divide task 3 into 4 separate functions:

1. bestvenue()
2. bestcaterer()
3. beststudio()
4. bestentertainer()

where each function has the same purpose - to pick the best vendor and simultaneously store their respective costs into variables that will be used in the final function when calculating the total expenditure.

VENUES

As aforementioned, tables were created to store details of various vendors for each expense. The table **Venues** includes details of the:

1. **name** of the venue
2. **city** it's located in
3. **type** of venue (hotel/park/resort etc.)
4. **space** (indoors/outdoors)
5. **capacity** - how many people it can host
6. **price** of the space

Field	Type	Null	Key	Default	Extra
Name	varchar(30)	NO		NULL	
City	varchar(30)	NO		NULL	
Type	varchar(30)	NO		NULL	
Space	varchar(30)	NO		NULL	
Capacity	int	NO		NULL	
Price	varchar(30)	YES		NULL	

Name	City	Type	Space	Capacity	Price
Radisson Pink	Chennai	Banquet Hall	Indoor	150	20L
Willow Hills	Bangalore	Hotel	Indoor	100	30L
The Boardwalk	Bangalore	Resort	Outdoor	50	10L
Malt & Cigar	Bangalore	Resto Lounge	Outdoor	75	5L
Aurelia's Gardenia	Chennai	Park with Gazebo	Outdoor	120	10L
The Pergola	Bangalore	Park with Gazebo	Outdoor	125	25L
Westend Inn	Chennai	Hotel	Indoor	150	40L
Luxuria Hall	Bangalore	Banquet Hall	Indoor	200	5L
The Lilac Orchid	Chennai	Resto Lounge	Outdoor	100	15L
Sterling Riviere	Chennai	Resort	Outdoor	50	10L

VENUES

With the details of the venues all ready, we define a function to get inputs from the user in order to pick the best venue:

- **Capacity filter**: Based on the total number of guests (as calculated in task 2), we automatically filter out the options and present to the user only those venues that can accommodate that number.

- **Location filter**: We ask the user for their preferred location and narrow down the possible venues further

- **Type filter**: The last filter is the type. We ask the user to pick a type and present to them the final venue

- **Confirmation step**: The user can either confirm, or choose to restart the process.

- **Storing cost**: The cost of the venue that's been confirmed is stored into a variable (*venue_cost*)

In our code, we use validation checks to ensure that inputs given to the program are as accurate as possible. We also employ the *format()* and *fetchall()* functions as shown.

The *bestvenue()* function:

```
# function3
def bestvenue(total_guests):
    priority = 1
    while priority == 1:

        # filtering out options based on capacity
        cursor.execute("select name from venues where capacity>%s" % total_guests)
        data = cursor.fetchall()
        available_venue = ()
        for row in data:
            available_venue += row
        print("Based on the total number of guests, these venues are available to you:", (list(available_venue)))

        # filtering out a venue based on location
        city = int(input("Enter which city you would like to have the wedding in. "
                         "\n Enter '1' for Chennai."
                         "\n Enter '2' for Bangalore"))
        if city == 1:
            city_string = "Chennai"
        elif city == 2:
            city_string = "Bangalore"

        # validation check
        if city != 1 and city != 2:
            print("Please check again."
                  "\n Enter '1' for Chennai"
                  "\n Enter '2' for Bangalore")

        elif city == 1:
            y = 'Chennai'
        elif city == 2:
            y = 'Bangalore'

        query = "select name from venues where city='" + y + "' and capacity>{}"
        cursor.execute(query.format(total_guests))
        data = cursor.fetchall()
        chennai_venues = ()
        for row in data:
            chennai_venues += row
        print("Your options are now", list((chennai_venues)))

        # filtering venue based on type
        type = int(input("Do you want an hotel, a banquet hall, a resto-lounge, or a resort? Enter:"
                         "\n 1: Hotel"
                         "\n 2: Banquet Hall"
                         "\n 3: Resto-Lounge"
                         "\n 4: Resort"
                         "\n 5: Park (Gazebo)"))
        # validation check
        if type != 1 and type != 2 and type != 3 and type != 4 and type != 5:
            int(input("Please check again. Enter:"
                      "\n 1: Hotel"
                      "\n 2: Banquet Hall"
                      "\n 3: Resto-Lounge"
                      "\n 4: Resort"
                      "\n 5: Park (Gazebo)"))
```

```
elif type == 1:
    x = 'hotel'
elif type == 2:
    x = 'banquet hall'
elif type == 3:
    x = 'resto lounge'
elif type == 4:
    x = 'resort'
elif type == 5:
    x = 'park with gazebo'
query1 = "select name,space,price from venues where type='" + x + "' and city='{}' and capacity>{}"

cursor.execute(query1.format(city_string, total_guests))
data = cursor.fetchall()

if len(data) == 0:
    print("There are no venues available. Restarting process.")

for row in data:

    print("Details of the only available venue:", row)
    confirm = input("Pick venue? Enter 'y' or 'n'")

    if confirm == 'y':
        venue = row[0]
        print("The venue you have picked is", venue,
              "\n It costs", row[2], "where L is Lakhs",
              "\n It is an", row[1], "space")

        str1 = str(row[2])
        if str1[1].isdigit():
            str2 = str1[0] + str1[1]
            venue_cost = int(str2) * 100000

            print("The cost for the venue amounts to", venue_cost, "rupees")
            return (venue_cost)
        quit()

    else:
        print("Restarting process.")
```

Thus, the first expense is settled.

CATERERS

Similarly, the table **Caterers** includes details of the:

1. **name** of the catering crew
2. **cuisine** (Indian/Chinese, Italian etc.)
3. **food type** (vegetarian/non vegetarian)
4. **capacity** - how many people can be served
5. **price per plate**

Field	Type	Null	Key	Default	Extra
Name	varchar(50)	YES		NULL	
Cuisine	varchar(50)	YES		NULL	
Food_Type	varchar(50)	YES		NULL	
Capacity	int	NO		NULL	
Price_per_plate	int	NO		NULL	

Name	Cuisine	Food_Type	Capacity	Price_per_plate
Red Box	Chinese	Non veg	200	500
The Tandoor Oven	Indian	Non veg	150	800
Asian Tales	Pan-asian	Non veg	125	600
Mexican Grill	Mexican	Non veg	100	400
Pasta la Vista	Italian	Veg	150	700
Nasi & Mee	Singaporean	Veg	115	1000
Paix, Amour & Bonheur	French	Veg	115	1000
Cherry wine & Sake	Japanese	Non veg	130	750
The Curry Kitchen	Indian	Non veg	130	850
Pho Kit	Thai	Non veg	140	650

CATERERS

With the details of the caterers all ready, we define a function to get inputs from the user in order to pick the best catering crew:

- **Capacity filter**: Based on the total number of guests (as calculated in task 2), we automatically filter out the options and present to the user only those catering crews that can serve that many people

- **Food Preference filter**: Based on the food preferences (whether they are vegetarian or non vegetarian) of each guest as stored in the guest list table, we automatically filter out the caterers and present to the user those crews that accommodate the majority of guests.

- **Cuisine filter**: The last filter is the cuisine. We ask the user for their preference and narrow down options.

- **Confirmation step**: The user can either confirm, or choose to restart the process.

- **Storing cost**: The cost of the caterer that's been confirmed is stored into a variable (*caterer_cost*)

The *bestcaterer()* function:

```
# function4:
def bestcaterer(total_guests, guestyesnum):
    priority = 2
    while priority == 2:

        # filtering out options based on capacity
        cursor.execute("select name from caterers where capacity>%s" % total_guests)
        data = cursor.fetchall()
        available_caterers = ()
        for row in data:
            available_caterers += row
        print("Based on the total number of guests, these venues are available to you:", (list(available_caterers)))

        # filtering out options based on food_preference
        cursor.execute("select count(food_preference) from guestlist where food_preference='non-vegetarian'")
        num = str(cursor.fetchall()[0])
        num1 = int((num[1] + num[2]))
        num2 = guestyesnum / 2

        if num1 > num2:

            food_type = 'non veg'
            cursor.execute("select name from caterers where food_type = 'non veg' and capacity>%s" % total_guests)
            data = cursor.fetchall()
            nonveg_caterers = ()
            for row in data:
                nonveg_caterers += row
            print("The majority of your guests prefer", food_type, "food")
            print("Based on this, your options are now:", list(nonveg_caterers))

        elif num1 < num2:

            food_type = 'veg'
            cursor.execute("select name from caterers where food_type = 'veg' and capacity>%s" % total_guests)
            data = cursor.fetchall()
            veg_caterers = ()
            for row in data:
                veg_caterers += row
            print("The majority of your guests prefer", food_type, "food")
            print("Based on the majority of people's food preference, your options are now:", list(veg_caterers))

        # filtering out options based on cuisine:
        cuisine = int(input("Pick a cuisine. You can choose from:"
                            "\n 1: Chinese"
                            "\n 2: Indian"
                            "\n 3: Pan-asian"
                            "\n 4: Mexican"
                            "\n 5: Italian"
                            "\n 6: Singaporean"
                            "\n 7: Japanese"
                            "\n 8: Thai"
                            "\n 9: French"))
```

```
if cuisine == 1:
    x = 'chinese'
elif cuisine == 2:
    x = 'indian'
elif cuisine == 3:
    x = 'pan-asian'
elif cuisine == 4:
    x = 'mexican'
elif cuisine == 5:
    x = 'italian'
elif cuisine == 6:
    x = 'singaporean'
elif cuisine == 7:
    x = 'japanese'
elif cuisine == 8:
    x = 'thai'
elif cuisine == 9:
    x = 'french'

query = "select name,price_per_plate from caterers where cuisine = '" + x + "' and capacity>{} and food_type='{}'"
cursor.execute(query.format(total_guests, food_type))
data = cursor.fetchall()

if len(data) == 0:
    print("There are no caterers that match your preferences available. Restarting process.")

for row in data:
    print("Details of the caterer(s) that satisfy your criteria:", row)
    confirm = input("Pick venue? Enter 'y' or 'n'")

    if confirm == 'y':
        caterer = row[0]
        print(row)
        print("The caterer you have picked is", caterer,
              "\n The cost per plate is", row[1], "rupees.",
              "\n It serves", x, "food.")

        caterer_cost = int(row[1]) * total_guests
        print("The catering cost amounts to", caterer_cost, "rupees")

        return caterer_cost

        quit()

    else:
        print("Restarting process.")
```

Thus, the second expense is settled.

STUDIOS

Similarly, the table **Photographers** includes details of the:

1. **name** of the studio
2. **rating** out of 5
3. **price**

Field	Type	Null	Key	Default	Extra
Name	varchar(50)	NO		NULL	
Price	varchar(10)	NO		NULL	
Rating	varchar(5)	NO		NULL	

Name	Price	Rating
Maya's Studio	40000	3.5
Faded Polaroids	70000	5.0
Portraits by Greyson Chance	40000	3.5
Photos & Frames	45000	4.0
Videos by Vivian	45000	4.0
Photography by Amelie	65000	4.5
High Definitions	50000	4.5
Pixels Lite	60000	4.5
Colours of the Wind	35000	4.0
CamerArt	30000	3.5

STUDIOS

With the details of the photographers all ready, we define a function to get inputs from the user in order to pick the best studio:

- **Rating filter**: We list all of the studios. Then, we ask the user to pick a rating. Based on whether they're okay with a studio that's rated a 3.5/5.0 or whether they specifically want one that's been rated higher, we list the available studios.

- **Confirmation step**: The available studios are presented in a list to the user. They can either confirm, or choose to restart the process.

- **Storing cost**: The cost of the studio that's been confirmed is stored into a variable (*photographer_cost*)

The *beststudio()* function:

```
# function5:
def beststudio():
    priority = 3
    while priority == 3:

        # listing out the studios
        cursor.execute("select name from photographers")
        data = cursor.fetchall()
        available_photographers = ()
        for row in data:
            available_photographers += row
        print("The following studios are available:", list(available_photographers))

        # filtering out options based on the rating
        rating = int(input("All of the studios listed have a rating of 3.5 and above out of 5. Pick a rating:"
                           "\n 1: = 3.5"
                           "\n 2: = 4.0"
                           "\n 3: = 4.5"
                           "\n 4: = 5.0"))
        if rating == 1:
            x = '3.5'
        elif rating == 2:
            x = '4.0'
        elif rating == 3:
            x = '4.5'
        elif rating == 4:
            x = '5.0'

        query = "select name,price from photographers where rating ='" + x + "'"
        cursor.execute(query.format(rating))
        data = cursor.fetchall()
        range_studios = []
        price_studios = []

        for row in data:
            range_studios.append(row[0])
            price_studios.append(row[1])
        print("Your options with a", x, "rating are:", range_studios, "and they cost", price_studios, "respectively")

        studio = int(input("Pick a studio. Enter the corresponding number."))
        choice = range_studios[studio - 1]
        print("You've picked", choice)
        confirm = input("Confirm? Enter 'y' or 'n'")

        if confirm == 'y':
            print("The studio you are hiring is", choice)
            print("It costs", price_studios[studio - 1], "rupees")

            photographer_cost = price_studios[studio - 1]
            return photographer_cost
            quit()

        else:
            print("Repeating process.")
```

Thus, the third expense is settled.

ENTERTAINMENT

Similarly, the table **Entertainment** includes details of the:

1. **name** of the entertainer
2. **type** of the entertainment (live band/comedian etc.)
3. **price** of the entertainment

Field	Type	Null	Key	Default	Extra
Name	varchar(50)	YES		NULL	
Type	varchar(50)	YES		NULL	
Price	int	YES		NULL	

Name	Type	Price
Nomadic Jams	Live Band	80000
Soulful Raga	Live Band	50000
Illusions	Magician	60000
Sabrina Carpenter	Solo Artist	100000
Kenny Sebastian	Comedian	80000
Aurelia West	Solo Artist	75000
Mr.Mike	Anchor	45000
Trivial Trivia	Games	65000
Bounce in your Step	Games	45000
Rupi Kaur	Comedian	55000

ENTERTAINMENT

With the details of the entertainers all ready, we define a function to get inputs from the user in order to pick the best entertainer:

- **Type filter:** We list the types of entertainment that are available. The user enters an option and we present to them the available entertainers based on their input.

- **Confirmation step**: The available entertainers are presented in a list to the user. They can either confirm, or choose to restart the process.

- **Storing cost**: The cost of the studio that's been confirmed is stored into a variable (*entertainment_cost*)

The *bestentertainer()* function:

```
# functions:
def bestentertainer():
    priority = 4
    while priority == 4:

        # listing out the entertainment
        cursor.execute("select name from entertainment")
        data = cursor.fetchall()
        available_entertainers = ()
        for row in data:
            available_entertainers += row
        print("The following entertainers are available:", list(available_entertainers))

        # filtering out options based on the rating
        e_type = int(input("Which type of entertainment are you looking for? Enter:"
                           "\n 1: Live Band"
                           "\n 2: Solo Artist"
                           "\n 3: Comedian"
                           "\n 4: Games"
                           "\n 5: Magician"))
        if e_type == 1:
            x = 'Live Band'
        elif e_type == 2:
            x = 'Solo Artist'
        elif e_type == 3:
            x = 'Comedian'
        elif e_type == 4:
            x = 'Games'
        elif e_type == 5:
            x = 'Magician'

        query = "select name,price from entertainment where type ='" + x + "'"
        cursor.execute(query)
        data = cursor.fetchall()
        entertainment_names = []
        entertainment_prices = []

        for row in list(data):
            entertainment_names.append(row[0])
            entertainment_prices.append(row[1])
        print("Your", x, "options are:", entertainment_names, "and they cost", entertainment_prices,
              "rupees respectively")

        entertainer = int(input("Pick an entertainer. Enter the corresponding number."))
        choice = entertainment_names[entertainer - 1]
        print("You've picked", choice)
        confirm = input("Confirm? Enter 'y' or 'n'")

        if confirm == 'y':
            print("The entertainment you are hiring is", choice)
            print("It costs", entertainment_prices[entertainer - 1], "rupees")

            entertainment_cost = entertainment_prices[entertainer - 1]
            return entertainment_cost
            quit()

        else:
            print("Repeating process.")
```

TASK 4

With the vendors confirmed, and the costs of each expense stored in their respective variables, all that's left to do is to add them up and calculate the total expenditure. In addition, the individual expenses are updated into the **Expenditure** table. Finally, we'll be subtracting the total expenditure from the assigned budget:

```
def expenditure(budget, venue_cost, caterer_cost, photographer_cost, entertainment_cost):
    # updating the expenditure table by inserting the expenses
    for y in ["venue", "caterer", "photographer", "entertainment"]:
        if y == "venue":
            x = venue_cost
        elif y == "caterer":
            x = caterer_cost
        elif y == "photographer":
            x = photographer_cost
        elif y == "entertainment":
            x = entertainment_cost

        expenditure_values = "update expenditure set cost = '" + str(x) + "' where expense = '" + y + "'"
        print(expenditure_values)
        cursor.execute(expenditure_values)
        mycon.commit()

    cursor.execute("select sum(cost) from expenditure")
    data = cursor.fetchall()
    total_expenditure = int(''.join(map(str, data[0])))
    print("The total expenditure amounts to", total_expenditure, "rupees")
    balance = budget - total_expenditure
    if balance > 0:
        print("You are well within your budget with a balance of", balance,
              "rupees. Congratulations on the wedding!")
        quit()

    elif balance == 0:
        print("The total expenditure of the wedding amounted to the same value as your budget.")
        quit()
    elif balance < 0:
        print("You have exceeded your budget by", (-1 * balance), "rupees.")
```

TASK 4

the **Expenditure** table before the program was run:

Expense	Cost
Venue	0
Caterer	0
Photographer	0
Entertainment	0

the **Expenditure** table after the vendors have been picked:

Expense	Cost
Venue	4000000
Caterer	82600
Photographer	45000
Entertainment	100000

MAIN PROGRAM

The main program that calls all the functions is as shown:

```
# mainprogram

print("Welcome to your one-stop wedding planning service! First, we'll be needing your guest list.")
guestlist()

print("We're now going to calculate how many people are coming to the wedding.")
guestyesnum, total_guests = totalguests()
budget = int(input("Perfect. Now, please enter a budget for the wedding."))

print("Time to pick a venue")
venue_cost = bestvenue(total_guests)

print("Moving on to hiring a caterer")
caterer_cost = bestcaterer(total_guests, guestyesnum)

print("The next step is to hire a photographer")
photographer_cost = beststudio()

print("Finally - pick a form of entertainment")
entertainment_cost = bestentertainer()

print("Cool. Calculating your total expenditure.")
expenditure(budget, venue_cost, caterer_cost, photographer_cost, entertainment_cost)
```

OUTPUT

creating the guest list:

```
Successfully connected to MySQL Database
Welcome to your one-stop wedding planning service! First, we'll be needing your guest list.
How many guests are you inviting?1
Enter the name of the guestMaya
Enter the guest's emailmayavarsh@gmail.com
Enter the guest's contact number917600475
Is the guest attending? Enter 'yes' or 'no'yes
Enter the number of additional people this guest is bringing3
insert into guestlist (Name, Number, RSVP, Additional_People,Email) values ('Maya',917600475,'yes',3,'mayavarsh@gmail.com')
Guest list successfully updated!
Do you want to see the guest list? Enter 'y' or 'n'n
```

calculating the total number of guests:

```
We're now going to calculate how many people are coming to the wedding.
The number of invitees who responded with 'yes' is 37
The number of additional guests is 81
The total number of attendees is 118
Do you want to see the updated guest list? Enter 'y' or 'n'n
Perfect. Now, please enter a budget for the wedding.85000000
```

picking the best venue:

```
Time to pick a venue
Based on the total number of guests, these venues are available to you: ['Radisson Pink', "Aurelia's Gardenia", 'The Pergola', 'Westend Inn', 'Luxuria Hall']
Enter which city you would like to have the wedding in.
 Enter '1' for Chennai.
 Enter '2' for Bangalore2
Your options are now ['The Pergola', 'Luxuria Hall']
Do you want an hotel, a banquet hall, a resto-lounge, or a resort? Enter:
 1: Hotel
 2: Banquet Hall
 3: Resto-Lounge
 4: Resort
 5: Park (Gazebo)1
There are no venues available. Restarting process.
Based on the total number of guests, these venues are available to you: ['Radisson Pink', "Aurelia's Gardenia", 'The Pergola', 'Westend Inn', 'Luxuria Hall']
Enter which city you would like to have the wedding in.
 Enter '1' for Chennai.
 Enter '2' for Bangalore1
Your options are now ['Radisson Pink', "Aurelia's Gardenia", 'Westend Inn']
Do you want an hotel, a banquet hall, a resto-lounge, or a resort? Enter:
 1: Hotel
 2: Banquet Hall
 3: Resto-Lounge
 4: Resort
 5: Park (Gazebo)1
Details of the only available venue: ('Westend Inn', 'Indoor', '40L')
Pick venue? Enter 'y' or 'n'y
The venue you have picked is Westend Inn
 It costs 40L where L is Lakhs
 It is an Indoor space
The cost for the venue amounts to 4000000 rupees
```

picking a caterer - no options available:

```
Moving on to hiring a caterer
Based on the total number of guests, these venues are available to you: ['Red Box', 'The Tandoor Oven', 'Asian Tales', 'Pasta La Vista', 'Cherry wine & Sake', 'The Curry Kitchen', 'Pho Kit'
The majority of your guests prefer veg food
Based on the majority of people's food preference, your options are now: ['Pasta La Vista']
Pick a cuisine. You can choose from:
 1: Chinese
 2: Indian
 3: Pan-asian
 4: Mexican
 5: Italian
 6: Singaporean
 7: Japanese
 8: Thai
 9: French6
There are no caterers that match your preferences available. Restarting process.
```

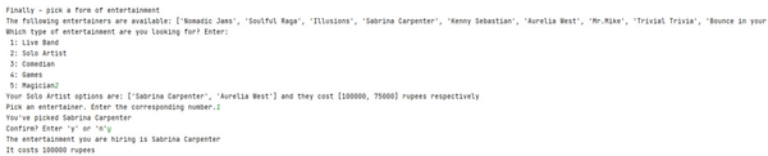
picking a caterer - confirming a crew:

```
Based on the total number of guests, these venues are available to you: ['Red Box', 'The Tandoor Oven', 'Asian Tales', 'Pasta La Vista', 'Cherry wine & Sake', 'The Curry Kitchen', 'Pho Kit
The majority of your guests prefer veg food
Based on the majority of people's food preference, your options are now: ['Pasta La Vista']
Pick a cuisine. You can choose from:
 1: Chinese
 2: Indian
 3: Pan-asian
 4: Mexican
 5: Italian
 6: Singaporean
 7: Japanese
 8: Thai
 9: French5
Details of the caterer(s) that satisfy your criteria: ('Pasta La Vista', 700)
Pick venue? Enter 'y' or 'n'y
('Pasta La Vista', 700)
The caterer you have picked is Pasta La Vista
 The cost per plate is 700 rupees.
 It serves italian food.
The catering cost amounts to 82600 rupees
```

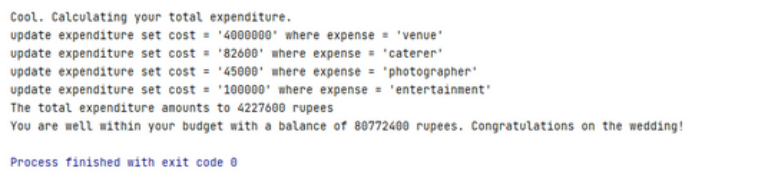
picking a studio:

```
The next step is to hire a photographer
The following studios are available: ["Maya's Studio", 'Faded Polaroids', 'Portraits by Greyson Chance', 'Photos & Frames', 'Videos by Vivian', 'Photography by Amelie', 'High Definitions',
All of the studios listed have a rating of 3.5 and above out of 5. Pick a rating:
 1: = 3.5
 2: = 4.0
 3: = 4.5
 4: = 5.03
Your options with a 4.5 rating are: ['Photography by Amelie', 'High Definitions', 'Pixels Lite'] and they cost ['65000', '50000', '60000'] respectively
Pick a studio. Enter the corresponding number.2
You've picked High Definitions
Confirm? Enter 'y' or 'n'n
Repeating process.
The following studios are available: ["Maya's Studio", 'Faded Polaroids', 'Portraits by Greyson Chance', 'Photos & Frames', 'Videos by Vivian', 'Photography by Amelie', 'High Definitions',
All of the studios listed have a rating of 3.5 and above out of 5. Pick a rating:
 1: = 3.5
 2: = 4.0
 3: = 4.5
 4: = 5.02
Your options with a 4.0 rating are: ['Photos & Frames', 'Videos by Vivian', 'Colours of the Wind'] and they cost ['45000', '45000', '35000'] respectively
Pick a studio. Enter the corresponding number.1
You've picked Photos & Frames
Confirm? Enter 'y' or 'n'y
The studio you are hiring is Photos & Frames
It costs 45000 rupees
```

picking entertainment::



calculating total expenditure:



Thus, we have successfully:

1. created tables
2. inserted values into a table
3. updated tables
4. altered tables
5. deleted redundant records from a table
6. employed validation checks
7. performed basic calculations

using Python and MySQL

USER MANUAL

DOWNLOADING PYTHON

https://www.anaconda.com/products/individual

To avoid having to manually install packages yourself, install Anaconda, and download your preferred Python interpreter (3.0 was used in this project). Create an environment file. Install the MySQL-connector package if needed. Run the code on your python interpreter. We used PyCharm Community Edition.

DOWNLOADING SQL

https://dev.mysql.com/downloads/installer/

From this link, download the MySQL application that suits your needs best (based on your operating system, and processor.) During the installation process, make sure that the Python connector is installed. For this project, we used MySQL Shell 8.0 - this application can be downloaded via the installation wizard if specified.

USER MANUAL

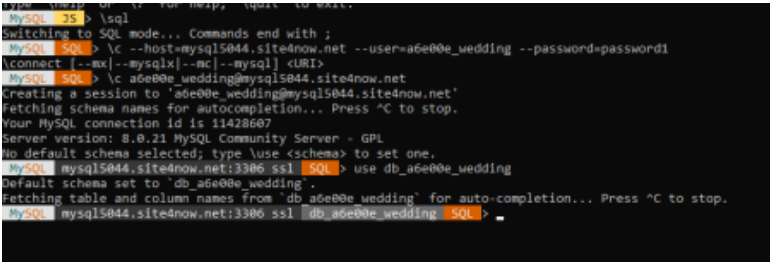
CONNECTING TO THE SERVER

https://www.smarterasp.net/

We stored our database on a free external hosting site. To connect via MySQL shell 8.0, use the following queries:

\c --host=[host] --user=[login ID}--password=[password]
\c [user]@[host]

our database password: password1
our database name: db_a6e00e_wedding
our database server: mysql5044.site4now.net
our user ID: a6e00e_wedding



REFERENCES

https://stackoverflow.com/questions/45465463/convert-tuple-to-int-in-python

https://www.geeksforgeeks.org/python-convert-tuple-string-to-integer-tuple/

https://www.geeksforgeeks.org/python-convert-tuple-to-integer/?ref=rp

https://member5.smarterasp.net/cp/cp_screen

https://www.w3schools.com/python/ref_string_format.asp

https://stackoverflow.com/questions/902408/how-to-use-variables-in-

https://towardsdatascience.com/a-simple-approach-to-templated-sql-queries-in-python-adc4f0dc511

https://www.programiz.com/python-programming/methods/list/remove

https://www.programiz.com/python-programming/function

REFERENCES

https://www.w3schools.com/sql/sql_like.asp
https://www.1keydata.com/sql/alter-table-rename-column.html

https://www.w3schools.com/sql/sql_null_values.asp

https://www.w3schools.com/sql/sql_alter.asp

https://www.wedmegood.com/vendors/all/wedding-entertainment/

https://www.candybar.co/blog/best-restaurant-names/

https://www.wedmegood.com/vendors/all/wedding-venues/all/hotel/

https://www.wedmegood.com/vendors/chennai/wedding-venues/

https://stackoverflow.com/questions/10718905/how-to-change-the-column-position-of-mysql-table-without-losing-column-data/10718926