Project Report

Dietician's Database

by

Krish Ahuja, Roll number: R002

Bhagyashree Birje, Roll number: R007

Tanishkaa Chaturvedi, Roll number: R011

Course: DBMS

AY: 2022-23

SVKM's NMIMS

Mukesh Patel School of Technology Management & Engineering A.Y. 2022 - 23

Course: Database Management Systems

Project Report

Program	MBA Tech Artificial Into	elligence
Semester	III	
Name of the Project:	Dietician's Database	
Details of Project Members		
Batch:	Roll No.	Name:
B1	R002	Krish Ahuja
B1	R007	Bhagyashree Birje
B1	R011	Tanishkaa Chaturvedi
Date of Submission: 19/10/2022		

Contribution of each project Members:

Roll No.	Name:	Contribution		
R002	Krish Ahuja	ERD, storyline, Relational model		
R007	Bhagyashree Birje	ERD, Debugging, normalization, queries		
R011	Tanishkaa Chaturvedi	ERD, table creation, data entry		

Table of Contents

Sr no.	Торіс	Page no.
	C. I.	2
1	Storyline	3
2	Components of Database Design	4-7
3	Entity Relationship Diagram	8
4	Relational Model	9
5	Normalization	10
6	SQL Queries	11-15
7	Project Demonstration	16-19
8	Learning from the Project	20
9	Challenges you faced while doing the Project	21
10	Conclusion	22

I. Storyline

The database addressed in our project is that of a Dietician healthcare agency that is run in two branches, Bandra and Vile Parle. The dieticians are allocated to the different branches and their fees are dependent on the experience of the dietician. The dietician agency has to coordinate with customers, dieticians, vendors and hospitals in order to successfully run their agency.

The dieticians are appointed and paid a salary based on their experience and they decide the meal chart of their customers. Customers experience different issues which can be helped by providing a diet suited for it. The vendors are those stores that will help to provide the meals as per their meal chart. We can run specific searches to provide the customers with vendors that are near their location.

The food groups are divided on the basis of the nutrient it specializes in and each consists of combinations of carbohydrates, vitamins, minerals, proteins and fibers. These food groups are then used to decide breakfast, lunch and dinner. The meal chart then consists of combinations of breakfast, lunch and dinner and is decided on the basis of the condition faced by the customer. The permutations and combinations of these are stored in the databases as per the suggestions of the dietician.

Before prescribing a particular diet, the customer has to undertake a blood test and also book an appointment with the dietician who will study these blood test results. Specific searches can be performed by the dietician to detect any anomalies in the customer's blood which helps to note the nutrients that they lack and can be compensated by the suggested meal plan.

The customer pays for the dietician services in installments which is recorded in a different table with the details of payment recorded duly.

II. Components of Database Design

Entities and attributes:

1.Customer (<u>customer_ID</u>, c_firstname, c_lastname, c_address,c_phone, c_age, blood group, <u>dietician_ID</u>, meal_no, pay_no)

The 'Customer' entity displays the customer ID, customer's first as well as last name, customer's address, customer's phone number, customer's age, customer's blood group with foreign keys that are dietician ID, their mean number, and their payment number. In the 'Customer' entity, the primary key is 'Customer_ID' which can display all the attributes of the customer entity present.

Eg. show* from customer where customer_ID=101;

2.Dietician (dietician_ID,issue_code, exp_code, salary, location)

The 'Dietician' entity displays the dietician ID, their salary, their location with foreign keys that are issue code and exp code. In the 'Dietician' entity, the primary key is the 'dietician_ID' which can display all the attributes of the dietician entity present.

Eg. show* from dietician where dietician_id=3;

3.lssue(<u>issue code</u>, issue desc)

The 'Issue' entity displays the issue code and the issue description. In the 'Issue' entity, the primary key is 'issue_code' which can display all the attributes of the issue entity present. The 'Issue' entity is related to the 'Customer' entity.

4.Experience(exp code, fees)

The 'Experience' entity displays the experiment code and the fees of the 'Dietician'. The primary key is 'exp_code' which can display all the attributes of the experience entity present. The 'Experiment' entity is related to the 'Dietician' entity.

5,Booking(dietappt_no, test_no, <u>customer_ID</u>)

The 'Booking' entity displays the diet appointment number, test number and has a foreign key- 'customer ID'. The primary key in the 'Booking' entity is customer_ID which is also the foreign key.

6.Dietbooking(<u>dietappt_no</u>, d_date,d_time)

The 'Dietbooking' entity displays the diet appointment number, diet booking date and diet booking time. The primary key is 'dietappt_no.' This is a weak entity which depends on the 'Booking' entity.

7.Testbooking(test_no, t_date ,t_time,hospital_no)

The 'Testbooking' entity displays the test number, test date, test time and the respective hospital number for the test to be conducted at. The primary key is 'test_no'. This is a weak entity which depends on the 'Booking' entity.

8.Blood_Test_result(<u>test_no</u>,platelets, hemoglobin, Urine test, Calcium)

The 'Blood test result' entity displays the platelets, hemoglobin, urine test, calcium with a foreign key which is also the primary key for this entity. This a weak entity which depends on the 'Testbooking' entity.

9.Meal chart(issue_code,meal_no, b_no, l_no, d_no)

The 'Meal chart' entity displays the meal number with the foreign keys: issue_code, breakfast_number, lunch_number, dinner_number. This entity depends on the 'Dietician' entity.

10.Breakfast(<u>b no</u>, **group_no**)

The 'Breakfast' entity displays the breakfast number with the foreign key 'group_no'. This depends on the 'Meal chart' entity.

11.Lunch(I no, group no)

The 'Lunch' entity displays the lunch number with the foreign key 'group_no'. This depends on the 'Meal chart' entity.

12.Dinner(<u>d no</u>, **group_no**)

The 'Dinner' entity displays the dinner number with the foreign key 'group_no'. This depends on the 'Meal chart' entity.

13. Food groups (group_no, group_name, m_no, v_no, p_no, c_no, f_no, group_desc, qty)

The 'Food groups' entity displays the group number, group name, group description, quantity with foreign keys 'minerals_no', 'vitamins_no', 'proteins_no', 'carbs_no', 'fibers_no'. The primary key is 'group_no'.

14.Minerals(m_no, m_name, m_desc)

The 'Minerals' entity displays the minerals number, minerals name and minerals description. The primary key is 'm_no'.

15. Vitamins (v_no, v_name, v_desc)

The 'Vitamins' entity displays the vitamins number, vitamins name and vitamins description. The primary key is 'v_no'.

16.Proteins(p_no, p_name, p_desc)

The 'Proteins' entity displays the protein number, protein name and protein description. The primary key is 'p_no'.

17.Carbs(c no, c name, c desc)

The 'Carbs' entity displays the carb number, carb name and carb description. The primary key is 'c no'.

18.Fibers(<u>f_no</u>, f_name, f_desc)

The 'Fibers' entity displays the fiber number, fiber name and fiber description. The primary key is 'f_no'.

19. Vendors(<u>Vendor_no</u>, vendor_name, v_contact, v_location, **group_no**)

The 'Vendors' entity displays the vendor number, vendor name, vendor contact, vendor location and the foreign key 'group no'. The primary key is 'vendor no'.

20.Hospitals(<u>hospital_no</u>, **issue_code**, h_location, h_phone)

The 'Hospital' entity displays the hospital number, hospital location, hospital phone with the foreign key 'issue_code'. The primary key is 'hospital_no'.

21.Payment(pay no,pay date, pay amt)

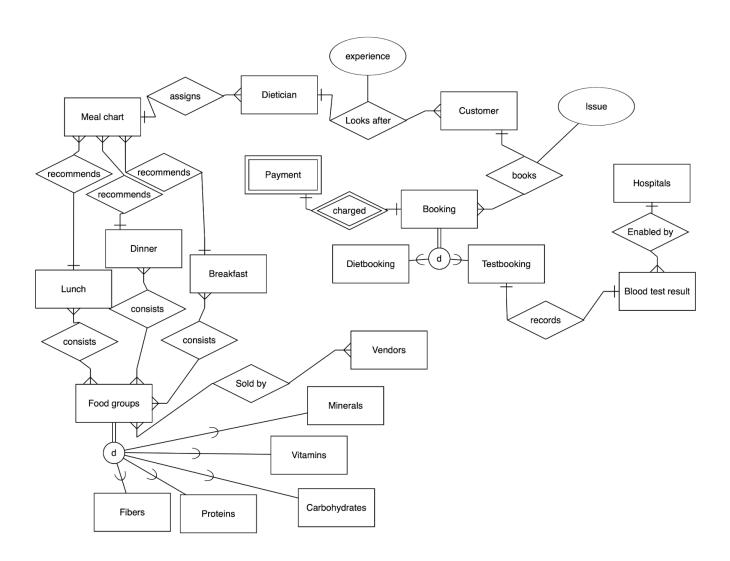
The 'Payment' entity displays the payment number, payment date, payment amount. The primary key is 'pay_no'.

Relationships:

- 1. Dietician looks after Customer. (one-to-many)
- 2. Dietician assigns Meal Chart. (many-to-many)
- 3. Customer books booking.(one-to-one)
- 4. Testbooking records Blood test result.(one-to-one)
- 5. Blood test result is enabled by Hospital.(many-to-one)
- 6. Meal chart recommends Lunch .(many-to-one)
- 7. Meal chart recommends Dinner.(many-to-one)
- 8. Meal chart recommends Breakfast.(many-to-one)
- 9. Lunch consists food groups.(many-to-one)

- 10. Dinner consists food groups.(many-to-one)
- 11. Breakfast consists food groups.(many-to-one)
- 12. Food groups is sold by Vendors .(many-to-many)

III. Entity Relationship Diagram



IV. Relational Model

In the following relational model, primary keys are underlined while the foreign keys have been made bold:

- 1. Customer (<u>customer_ID</u>, c_firstname, c_lastname, c_address,c_phone, c_age, blood group, <u>dietician_ID</u>, meal_no,pay_no)
- 2. Dietician(dietician ID, issue code, exp_code, salary, location)
- 3. lssue(<u>issue_code</u>, issue_desc)
- 4. Experience(<u>exp_code</u>, fees)
- 5. Booking(booking id, dietappt_no, test_no, customer ID)
- 6. Dietbooking(dietappt no, d date,d time)
- 7. Testbooking(test_no, t_date ,t_time, hospital_no)
- 8. Blood Test result(test_no,platelets, hemoglobin, Urine test, Calcium)
- 9. Meal chart(issue_code,meal_no, b_no, l_no, d_no)
- 10. Breakfast(<u>b_no</u>, **group_no**)
- 11. Lunch(I no, group no)
- 12. Dinner(<u>d</u> no, **group_no**)
- 13. Food groups (group_no, group_name,m_no, v_no, p_no, c_no, f_no, group_desc, qty)
- 14. Minerals(m_no, m_name, m_desc)
- 15. Vitamins (v_no, v_name, v_desc)
- 16. Proteins(p_no, p_name, p_desc)
- 17. Carbs(c_no, c_name, c_desc)
- 18. Fibers(f_no, f_name, f_desc)
- 19. Vendors(<u>Vendor_no</u>, vendor_name, v_contact, v_location, group_no)
- 20. Hospitals(hospital no, issue code, h location, h phone)
- 21. Payment(pay no,pay date, pay amt, customer id)

V. Normalization

Let us consider the table 'Customer' to be normalized:-

1NF:

For 1 Normal Form, the following requirements need to be achieved:

- 1. Each table has a primary key
- 2. The values in each column of a table are atomic
- 3. There are no repeating groups

Earlier the first table was customer with the following relational model:

Customer (<u>customer_ID</u>, <u>c_name</u>, c_address,c_phone, c_age, blood group, <u>dietician_ID</u>, <u>meal_no,pay_no,issue_code</u>)

In this the c_name was a multi-valued attribute, that is it had two values, first name and last name. So after applying 1NF, we divided the name attribute further into 2 attributes: first name and last name as follows:

Customer (<u>customer_ID</u>, <u>c_firstname</u>, <u>c_lastname</u>, <u>c_address</u>, <u>c_address</u>, <u>c_age</u>, blood group, <u>dietician_ID</u>, <u>meal_no</u>, <u>pay_no</u>, <u>issue_code</u>)

2NF:

For 2 Normal Form, the following requirements need to be achieved:

- 1. The table should be in 1NF
- 2. The table should have no partial dependencies

Then further in this table, the foreign key attribute issue_code was determining dietician_ID which is a subset of the primary key:

Customer (<u>customer_ID</u>,c_firstname, c_lastname, c_address,c_phone, c_age, blood group, <u>dietician_ID</u>, <u>meal_no,pay_no,issue_code</u>)

This means issue_code displayed a partial dependency on dietician_ID. Thus in order to eliminate this, a new table for issues was formed:

Issue(issue code, issue desc)

The customer table finally had these attributes:

Customer (<u>customer_ID</u>,c_firstname, c_lastname, c_address,c_phone, c_age, blood group, <u>dietician_ID</u>, <u>meal_no,pay_no</u>)

3NF:

For 3 Normal Form, the following requirements need to be achieved:

- 3. The table should be in 2NF
- 4. The table should have no transitive dependencies

There does not exist a transitive dependency in the customer table, thus we can say the customer table is already in 3NF and BCNF form.

VI. SQL Queries

Q.1 Show the details of the customer ID 1111 except the customer's age.

Code: select customer_id, c_firstname, c_lastname, c_address,c_phone,blood_group,dieticiamysql> select customer_id, c_firstname, c_lastname, c_address,c_phone,blood_group,dietician_id, meal_no, pay_no from customer where customer_id=1111;

Output:

İ	customer_id	c_firstname	 c_lastname	c_address	 c_phone	 blood_group	dietician_id	 meal_no	+ pay_no
	1111	Tanishkaa	Chaturvedi	402, Chakala	8764859324	B+	1002	12	2

Q.2 Show the most experienced dietician available

Code:

select * from dietician natural join experience where exp code=(select max(exp code) from experience);

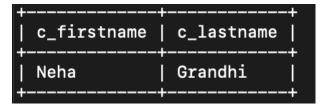
Output:

exp_code	+ dietician_id +	 salary	 issue_code	location	+ fees
4 4 4 4	1001 1004 1007 1009	70000 70000 60000 70000	1040 1010	Vile Parle Vile Parle Bandra Vile Parle	50000 50000 50000 50000

Q.3 Display customer's name where diet appointment number is 3.

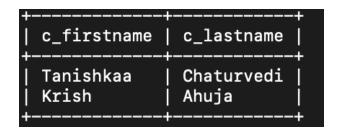
Code: select c_firstname, c_lastname from customer natural join dietbooking natural join booking where dietappt_no=3;

Output:



Q.4 Show the customer's names whose hemoglobin is greater than 1.

Code: select c_firstname, c_lastname from blood_test_result natural join customer natural join booking where hemoglobin>1;



Q.5 Update the timing for test booking number 3 to 5:00 Code: update testbooking set t time="5:00" where test no=3;

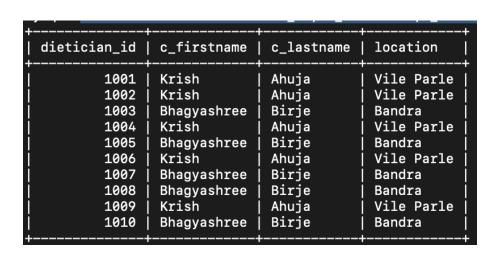
Output:

test_no			
	t_date	t_time	hospital_no
1		03:00:00	803
2		04:00:00	804
3		05:00:00	804
4		12:00:00	800

Q.6 Display the dietician id and customer name and location for those dieticians and customers who are around each other.

Code:

select dietician.dietician_id, c_firstname,c_lastname, location from dietician, customer where location=c_address;



Q.7 Sort the food groups on basis of quantity

Code:

select * from food_groups order by qty desc;

Output:

group_name	group_desc	qty	m_no	v_no	p_no	c_no	f_no
Sattvic food	Foods for clean and toxin free body	6	94	63	51	70	31
Vitamin packed non-veg	Healthy foods enriched with vitamins and minerals	3	92	64	j 54 j	72	30
Vitamin packed veg	healthy veg food enrichd with vitamins and minerals	3	94	63	j 50 j	70	j 31
Iron man	Iron-rich food for anemia and other similar conditions	3	94	63	j 50 j	70	j 31
Carbohydrate rich	Food for higher energy and carbohydrates	3	94	63	50 j	70	31
Calcium rich	Calcium rich food with necessary supplements for healthy bones	2	j 91 j	61	j 53 j	70	j 30 i
\ \ \ \ \	Vitamin packed non-veg Vitamin packed veg Iron man Carbohydrate rich	Vitamin packed non-veg Healthy foods enriched with vitamins and minerals Vitamin packed veg healthy veg food enrichd with vitamins and minerals Iron man Iron-rich food for anemia and other similar conditions Carbohydrate rich Food for higher energy and carbohydrates	Vitamin packed non-veg Healthy foods enriched with vitamins and minerals 3 Vitamin packed veg healthy veg food enrichd with vitamins and minerals 3 Iron man Iron-rich food for anemia and other similar conditions 3 Carbohydrate rich Food for higher energy and carbohydrates 3	Vitamin packed non-veg Healthy foods enriched with vitamins and minerals 3 92 Vitamin packed veg healthy veg food enrichd with vitamins and minerals 3 94 1 1 1 1 1 1 1 1 1	Vitamin packed non-veg Healthy foods enriched with vitamins and minerals 3 92 64 Vitamin packed veg healthy veg food enrichd with vitamins and minerals 3 94 63 Iron man Iron-rich food for anemia and other similar conditions 3 94 63 Carbohydrate rich Food for higher energy and carbohydrates 3 94 63	Vitamin packed non-veg Healthy foods enriched with vitamins and minerals 3 92 64 54 Vitamin packed veg healthy veg food enrichd with vitamins and minerals 3 94 63 50 Iron man Iron-rich food for anemia and other similar conditions 3 94 63 50 Carbohydrate rich Food for higher energy and carbohydrates 3 94 63 50	Vitamin packed non-veg Healthy foods enriched with vitamins and minerals 3 92 64 54 72 vitamin packed veg healthy veg food enrichd with vitamins and minerals 3 94 63 50 70 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

Q.8 Show those protein that have plant-based options Code:select * from proteins where p_name like "%plant%";

Output:

+	+	
p_no	p_name	p_desc
51 52	plant protein legume plant protein nuts plant protein grain plant protein veggies	lentils, beans, peas, soybean almonds,cashews,walnuts,sunflower seeds, sesame, chia rice, quinoa, oats, millet corn, broccoli, asparagus, sprouts

Q.9 Display dinner options for customer wishing to gain weight Code:

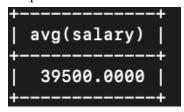
select m_desc, v_desc, p_desc,c_desc,f_desc,issue_code from meal_chart natural join dinner natural join food_groups natural join vitamins natural join minerals natural join proteins natural join carbs natural join fibers where issue_code=1020;

Output:

m_desc	v_desc	p_desc	c_desc	f_desc	issue_code
	Grains,broccoli, fruits, fish,milk,cheese Green leafy vegetables, turnip, beet green		pumpkin, carrot, tomatoes, beans, broccoli, cucumber apples, oranges, banana,pineapple,berries, milk	oatmeal,chia,lentils,apples,blueberries whole wheat, quinoa, brown rice, leafy greens, walnuts, seeds, pear, apples	1020

Q.10 What is the average salary for the dieticians Code: select avg(salary) from dietician;

Output:

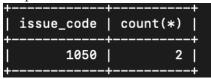


Q.11 What is the number of meal options available for a kid

Code:

select issue code,count(*) from meal chart group by issue code having issue code=1050;

Output:



Q.12 Create a view for the hospital displaying only test booking details and customer details

Code:

create view vwhospi as select test_no, customer_id,

t_date,t_time,hospital_no,h_location,h_phone,issue_code from booking natural join customer natural join testbooking natural join hospital;

Output:

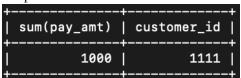
+	+	+	·			·	++
test_no	customer_id	t_date	t_time	hospital_no	h_location	h_phone	issue_code
1	 1111	2022-10-10	03:00:00	803	 Kanjurmarg	 6478352672	
2	2222	2022-10-10	04:00:00	804	Malad	675432685	1010
3	4444	2022-10-11	05:00:00	804	Malad	675432685	1010
j 4	3333	2022-10-12	12:00:00	800	Andheri East	9876543672	1030
+	+	<u> </u>				+	++

Q.13 Display the total amount paid by customer 1111

Code:

select sum(pay amt), customer id from customer natural join payment where customer id=1111;

Output:



Q.14 Show carbohydrate options with beans in it

Code:

select * from carbs where c desc like "%beans%";

++ c_no c_name	
	whole grains, grain bread, beans, potatoes, cereals pumpkin, carrot, tomatoes, beans, broccoli, cucumber

Q.15 Create a vendor view and join the tables visible to both the vendors as well as the hospital Code:

select * from vwhospi natural join vwvendor1;

issue_code	test_no	customer_id	t_date	t_time	 hospital_no	h_location	h_phone	vendor_name	v_location	vendor_no	group_name	meal_no
1020	1	1111	2022-10-10	03:00:00	803	Kanjurmarg	6478352672	Delomite	Andheri West	2222	Calcium rich	12
1020	1	1111	2022-10-10	03:00:00	j 803 j	Kanjurmarg	6478352672	Delomite	Andheri West	2222	Calcium rich	j 19 j
1010	2	2222	2022-10-10	04:00:00	804	Malad	675432685	Nageshwar	Bandra	3333	Sattvic food	j 10 j
1010	2	2222	2022-10-10	04:00:00	804	Malad	675432685	Kinara	Bandra	5555	Sattvic food	j 10 j
1010	2	2222	2022-10-10	04:00:00	804	Malad	675432685	Ashapura	Vile Parle West	1111	Iron man	11
1010	2	2222	2022-10-10	04:00:00	804	Malad	675432685	Onega	Bhandup	4444	Iron man	11
1010	3	4444	2022-10-11	05:00:00	804	Malad	675432685	Nageshwar	Bandra	3333	Sattvic food	10
1010	3	4444	2022-10-11	05:00:00	804	Malad	675432685	Kinara	Bandra	5555	Sattvic food	j 10 j
1010	3	4444	2022-10-11	05:00:00	804	Malad	675432685	Ashapura	Vile Parle West	1111	Iron man	11
1010	3	4444	2022-10-11	05:00:00	804	Malad	675432685	Onega	Bhandup	4444	Iron man	11
1030	4	3333	2022-10-12	12:00:00	800	Andheri East	9876543672	Nageshwar	Bandra	3333	Sattvic food	j 13 j
1030	4	3333	2022-10-12	12:00:00	800	Andheri East	9876543672	Kinara	Bandra	5555	Sattvic food	13
+	·		·	·	+			+	+	+	+	+

VI. Project demonstration

Our code was executed on the terminal and mysqldump was used to export the .sql file provided with this report.

Thus the format of the .sql file might appear differently on different softwares for the same code:

```
MySQL code:
create table payment(pay no int primary key,
pay date date,
pay_amt int );
create table minerals(m no int primary key,
m name varchar(100),
m desc varchar(300));
create table vitamins(v no int primary key, v name varchar(1000), v desc varchar(3000));
create table proteins(p no int primary key, p name varchar(1000), p desc varchar(3000));
create table carbs(c no int primary key, c name varchar (1000), c desc varchar(3000));
create table fibers(f no int primary key, f name varchar(1000), f desc varchar(3000));
create table food groups(group no int primary key,
group name varchar(30),
group_desc varchar(1000),
qty int,
m no int,
v_no int,
p no int,
c_no int,
f no int,
foreign key (m no) references minerals(m no),
foreign key (v_no) references vitamins(v_no),
foreign key (p no) references proteins(p no),
foreign key (c no) references carbs(c no),
foreign key (f_no) references fibers(f_no)
);
create table vendors(vendor no int primary key,
vendor name varchar (30),
```

```
v contact int,
v_location varchar(50),
group no int,
foreign key(group_no) references food_groups(group_no)
);
create table breakfast(
b_no int primary key,
group no int,
foreign key (group_no) references food_groups(group_no)
);
create table lunch(
I_no int primary key,
group_no int,
foreign key (group no) references food groups(group no)
);
create table dinner(
d_no int primary key,
group no int,
foreign key (group_no) references food_groups(group_no)
);
create table issue
(issue_code int primary key,
issue_desc varchar(30));
create table meal_chart(
issue_code int,
meal_no int primary key,
b_no int,
I_no int,
d_no int,
foreign key(issue code) references issue(issue code),
foreign key(b_no) references breakfast(b_no),
foreign key(l_no) references lunch(l_no),
foreign key(d no) references dinner(d no)
);
create table experience (exp_code int primary key, fees int);
```

```
create table dietician
(dietician id int,
exp_code int,
salary int,
issue code int,
location varchar(50),
primary key (dietician_id,issue_code),
foreign key (issue_code) references issue (issue_code),
foreign key(exp code) references experience(exp code)
);
create table dietbooking(dietappt no int primary key, d date date, d time time);
create table hospital(
hospital_no int primary key,
issue_code int,
h location varchar(30),
h_phone varchar(10),
foreign key (issue code) references issue (issue code)
);
create table testbooking (test_no int, int primary key, t_date date, t_time time,hospital_no int,
foreign key(hospital no) references hospital(hospital no);
create table customer
(customer_id int,
c firstname varchar(30),
c lastname varchar(30),
c_address varchar (50),
C phone varchar(10),
c age int,
blood_group varchar(10),
dietician id int not null,
meal_no int not null,
pay_no int,
primary key (customer_id,dietician_id),
foreign key(dietician id) references dietician(dietician id),
foreign key(meal_no) references meal_chart(meal_no),
foreign key(pay_no) references payment(pay_no)
);
create table booking
```

```
dietappt_no int,
test_no int,
customer_id int primary key,
foreign key(customer_id) references customer(customer_id),
foreign key(dietappt_no) references dietbooking(dietappt_no),
foreign key(test_no) references testbooking(test_no)
);

create table blood_test_result(test_no int primary key,
platelets int,
hemoglobin int,
urine_test int,
calcium int,
foreign key(test_no) references testbooking (test_no)
);
```

VII. Learning from the Project

• How did this project help you?

This project helped us in providing an interface to perform various operations like database creation, storing data in it, updating data, creating a table in the database and a lot more. We learnt different types of attributes and weak entities an entity has and the relations the entity could have. As per our research, we also got to know about different types of queries which also include the complicated queries to run efficiently. This also helps in understanding the input user (customer) gives and based on that our database works smoothly while keeping all the requirements by the customer in mind and giving the best possible result. In the case of multiple users, it also maintains data consistency. It also helped us to develop the analytical skills of planning any dataset given to us.

• What new aspects did you learn?

We learnt how to design and maintain business databases. One of the most important aspect we learnt is the use of different kinds of entities, attributes and their relations being displayed in the desired way as per the customer's need. The management between all the tables (entities) and all their relations including their tuples (attributes) helps us to make a more simple user friendly database. As all the tables have relationships either being (one-to-one, one-to-many, many-to-one or many-to-many), they have a complete balance which calculates all the possibilities and data given by the customer which would then help the database in giving the best possible result, as in this database to the customer. Using more tuples, more values, different relationships and depending on what type and also as complicated queries can be run shows what different aspects we learnt through this project.

VIII. Challenges Faced

During the implementation of the database, there were challenges during updation of the tables since the referential integrity needed to be maintained.

Eg:

After creating the table 'customer', we performed normalization and realized the foreign key attribute 'issue_code' needs to be removed from 'customer' table.

Inorder to do that, the command 'alter table customer drop issue_code' did not work. After researching on the same, we learnt that when dropping a foreign key attribute we first need to drop the constraint on it. So to know the constraint, we used the command 'show create table 'customer';'

This displayed the constraint name as 'customer_ibfk_1' for issue_code and so then we dropped the constraint first by using the command 'alter table customer drop customer_ibfk_1;' and then we ran the command 'alter table customer drop issue_code;'

During the insertion of data we faced certain challenges pertaining to either the data format or the type of "" used

Eg: For the testbooking and dietbooking tables, we had to accept dates in date format. So for the date format, there was an error which was resolved by using the command

'STR_TO_DATE("10-10-22","%d-%m-%y")' to insert the data value according to our desired database format. Furthermore during insertion of data the type of quotations used by the code also made a difference to the success of the sql code. The usage of "instead of "threw an error since the code surprisingly did not accept using ".

Extra care was required to ensure the table exists for the foreign keys before they are being inserted in the table being created.

Eg: The table 'dietician(dietician_ID, **issue_code**, **exp_code**, salary,location)' cannot be created before the tables that have 'issue_code' and 'exp_code' as primary keys have been created.

Overall debugging the code to ensure that all data inserted matches the value required was a crucial and time consuming task.

IX. Conclusion

The implementation of the diet agency database helped us to understand the components and their uses in a database more clearly and efficiently. It was an opportunity to reflect on the need for a database in real life as well. The following were our key takeaways from the project:

- 1. The better and more time is spent in planning of the database, the easier and faster it is to execute the database at the end.
- 2. There are several advantages for creating a database for any particular dataset like:
 - a. Saves time to run searches throughout the entire dataset
 - b. Helps to prevent anomalies during the insertion, deletion or updation of any database.
 - c. The data can be easily shared across to different people through the sql files.